VTB Visual Tool Basic

www.promax.it

Guida Agli Oggetti

PROMAX Motion & Control

Rev. 3.00.0

1 PREFAZIONE

Questo manuale è una guida agli oggetti di VTB,
Gli oggetti rappresentano una caratteristica importante del linguaggio di programmazione e semplificano lo sviluppo delle applicazioni. Esistono oggetti e funzioni tecnologiche di vario tipo adatti a risolvere specifiche situazioni.

2 CLASSE INPUTBIT

La Macro classe InputBit contiene oggetti relativi all' utilizzo delle variabili bit, sia associate a normali variabili, sia associate ad input digitali. Di conseguenza è possibile gestire lo stato dei Bit in modo semplice ed immediato rilevandone fronti di salita e di discesa utilizzando gli eventi già preparati.

2.1 CstdBit.vco – Gestione dei BIT

Classe che contiene oggetti che non hanno nessuna grafica di visualizzazione. Generano solamente due eventi **StatoOn** e **StatoOff** dove è possibile inserire codice di controllo.

Hardware ***Tutti***

Proprietà principali

Nome	Nome univoco dell' oggetto. Non gestibile in run time
Variabile	Nome della variabile bit su cui operare. Non gestibile in run time
Enable	NomeOggetto.Enable = True abilita gli eventi NomeOggetto.Enable = False disabilita gli eventi

Metodi

Nessuno

Eventi

StatoOn	Si verifica quando il bit passa allo stato logico 1, fronte di salita
StatoOff	Si verifica quando il bit passa allo stato logico 0, fronte di discesa

3 CLASSE MOTOR CONTROL

La Macro classe MotorControl include oggetti che si occupano del controllo della movimentazione assi. In genere il controllo della movimentazione assi prevede diverse tipologie specifiche (alberi elettrici, CAM, Posizionatori ecc.) che sono raccolte interamente in questa macro classe. VTB controlla periferiche esterne che rispondono allo standard CAN OPEN profilo DS301 e DSP402, STEP/DIR, Analogica +/-10V o Ethercat, quindi in linea di massima tutti i componenti che utilizzano questi protocolli, possono essere interamente gestiti. Una notevole facilitazione deriva dal fatto di poter utilizzare oggetti già predisposti e collaudati, questo non preclude comunque di poter utilizzare periferiche al di fuori della libreria oggetti fornita.

3.1 CbitCam.vco – Gestione CAMME a bit ON/OFF

Classe riferita a CAMME che vengono generate attivando un uscita digitale ad un certo valore di una quota MASTER e disattivandola ad un altro valore. Questa risulta utile per la gestione di PISTONI, sincronizzando il movimento con il MASTER.
Il MASTER può essere qualsiasi grandezza numerica disponibile (Encoder esterno, Asse esterno, asse virtuale ecc.)
Unica limitazione, è che la grandezza MASTER deve essere necessariamente un multiplo BINARIO:
512,1024,2048,4096 ecc.

Hardware ***Tutti***

Proprietà principali

Impulsi/giro	Impulsi a giro della sorgente master. **ATTENZIONE questo valore deve essere necessariamente BINARIO**
MasterSet	Valori in impulsi del master relativo al SET della variabile.BIT
MastertRes	Valori in impulsi del master relativo al RESET della variabile.BIT
Master	Variabile Sorgente che definisce il MASTER. Non gestibile in run time
VarBit	Variabile destinazione che puo' essere un bit che generalmente è associato ad un uscita digitale. Non gestibile in run time

Metodi

Enable	Nomeoggetto.Enable=True abilita il controllo del bit Nomeoggetto.Enable=False disabilita il controllo del Bit

Eventi

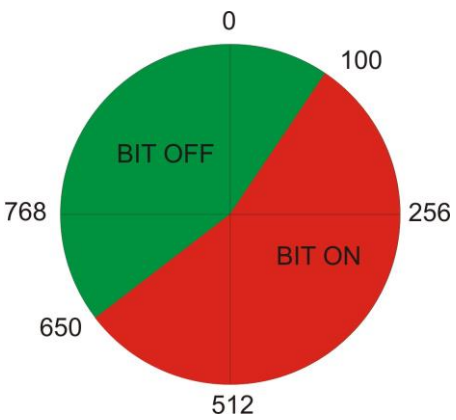
Nessuno

Esempio:

Valori impostati

Impulsi/giro=1024	impulsi a giro della sorgente master
MasterSet=100	Set Bit
MasterRes=650	Reset Bit

Risultato



3.2 Ccam.vco Camma /Camma Continua - Gestione CAMME elettroniche per motori

Classe riferita a CAMME ELETTRONICHE generiche create con strumenti esterni (CAD, ecc.). i tipi di CAMME possono essere continui (Camma Continua) o con ritorno a zero (Camma). Questa classe lavora essenzialmente su un vettore che deve avere una dimensione uguale agli impulsi di un MASTER. Quest' ultimo può essere qualsiasi fonte esterna od interna a alla piattaforma hardare (ENCODER, CONTATORE, VARIABILE ecc.) Ad ogni posizione del vettore corrisponde la relativa quota che deve portarsi l' asse SLAVE. In pratica l' encoder MASTER viene preso come INDICE DEL VETTORE che di conseguenza corrisponde alla relativa quota dello slave. La posizione dello SLAVE viene posta in una variabile che può essere associata ad un PDO, filtro PID, quota STEP.

Camma Continua Definisce un asse SLAVE che và sempre nella stessa direzione.
Camma con ritorno a zero definisce un asse SLAVE che parte da una quota (ZERO) e ritorna a questa

Hardware ***Tutti***

Proprietà principali

Nome	Nome univoco dell' oggetto. Non gestibile in run time
N. Punti	Numero punti che compongono la CAMMA, corrispondono al numero impulsi del MASTER. Non gestibile in run time
Master	Variabile Sorgente che definisce il MASTER. Non gestibile in run time
Slave	Variabile Destinazione del contenuto del vettore deve necessariamente corrispondere ad Non gestibile in run time
VetCam	Vettore che contiene i punti della CAMMA. Non gestibile in run time
Rotazione	Inverte la rotazione del motore rispetto al profilo disegnato Valori ammessi 0 o 1

Metodi

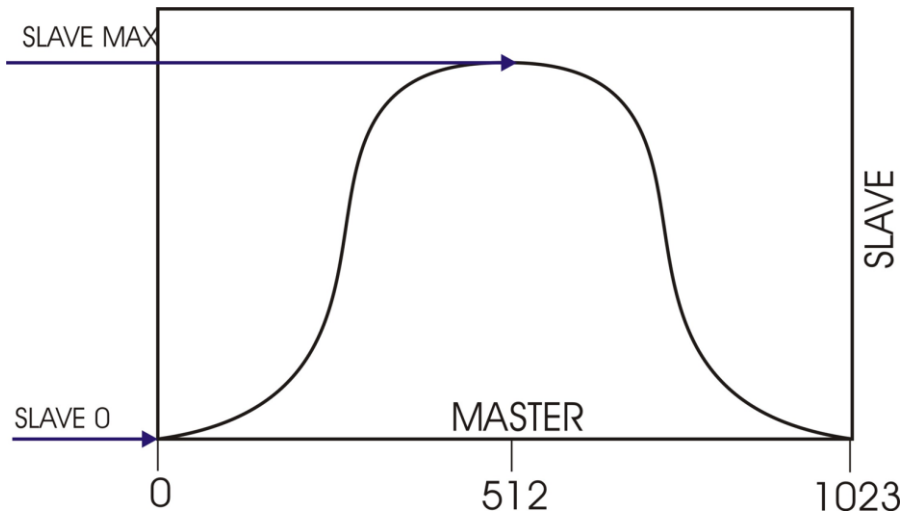
Enable	Nomeoggetto.Enable=True abilita il controllo del motore slave con il profilo CAM Nomeoggetto.Enable=False disabilita il controllo del motore
Fase	Nomeoggetto.Fase=Valore Indica la fase di partenza della CAMMA. In pratica definisce la partenza dalla posizione del vettore indicata in .fase

Eventi

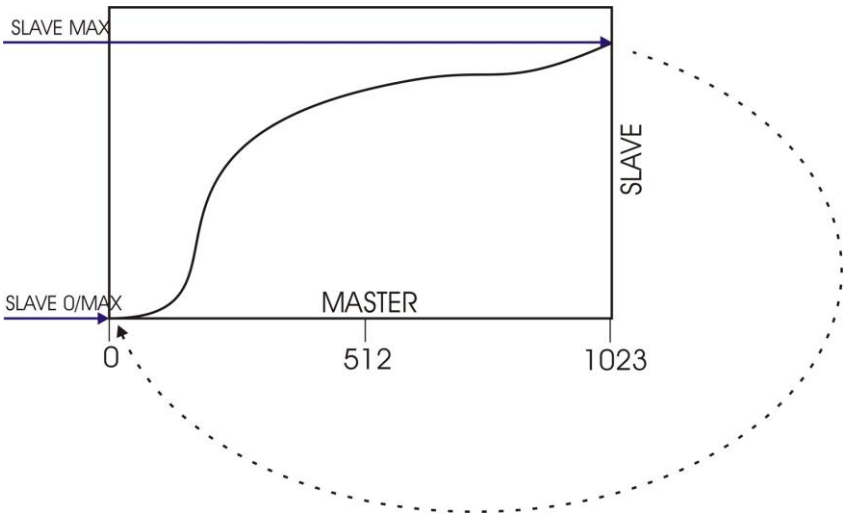
Nessuno

Per chiarezza viene disegnato un esempio di CAM che simula un master da 0 a 1024 impulsii, continua, cioè senza ritorno a zero, e con ritorno a zero

CAMMA CON RITORNO A ZERO



CAMMA CONTINUA



Esempio
Questo esempio rappresenta il codice per una CAMMA CON RITORNO A ZERO generica Camma1

Variabili Globali

MasterEnc	Long	Sorgente master
SlaveMot	Long	Destinazione Slave
VetCam(20)	Long	Vettore della CAMMA

Proprietà oggetto Camma1
Numero Punti=20
Master=MasterEnc
Slave=SlaveMot
VetCam=VetCam

```
' inizializzare nella MAIN VetCam
vcam1(0)=0
vcam1(1)=1
vcam1(2)=2
vcam1(3)=3
vcam1(4)=4
vcam1(5)=5
vcam1(6)=6
vcam1(7)=7
vcam1(8)=8
vcam1(9)=9
vcam1(10)=10
vcam1(11)=9
vcam1(12)=8
vcam1(13)=7
vcam1(14)=6
vcam1(15)=5
vcam1(16)=4
vcam1(17)=3
vcam1(18)=2
vcam1(19)=1
'Abilitare la CAM quando necessario
Camma1.Enable=True              ‘ Abilita il controllo della CAMMA
' ciclo nel task plc a 2 Ms
Inc MasterEnc
```

La variabile SlaveMot segue il profilo della VETCAM . Inizialmente il parametro FASE è a zero ciò vuol dire che alla posizione ZERO del MASTERENC corrisponde la posizione di VETCAM(0). Cambiando il valore della FASE è possibile cambiare il punto di inizio della CAMMA in riferimento dello ZERO del MASTER. L' esempio è puramente indicativo, in quanto sia i valori sia il numero di punti non sono sufficienti a generare un profilo continuo dell' asse slave.

3.3 CcamPulse.vco – Gestione CAMME a bit impulsive

Classe riferita a CAMME che vengono generate attivando un uscita digitale impulsiva per un durata impostabile. Ad un certo valore di MASTER maggiore o uguale a FASE, viene attivato il bit VarBit per il tempo DURATA. La variabile MASTER deve essere gestita esternamente, cioè il suo valore deve essere resettato (portato a zero) da esterno per poter riattivare la gestione del bit. Inoltre per avere un effettiva attivazione deve anche essere resettato il bit di FATTO.
Di fatto la Variabile MASTER è in valore che parte da un valore 0 e comunque ritorna a questo valore.
In caso di assi rotativi occorre riportare un asse virtuale da 0 a N.

Hardware ***Tutti***

Proprietà principali

Fase	Valori in impulsi del master relativo al SET della variabile BIT.
Durata	Durata in Millisecondi di SET della variabile BIT.
Master	Variabile Sorgente che definisce il MASTER. Deve essere resettata da esterno.
VarBit	Variabile Destinazione che può essere un bit che generalmente è associato ad un uscita digitale. Non gestibile in run time
Fatto	Viene settato dall' oggetto quando ha effettuato il set della variabile, deve essere resettato da esterno per un effettivo reload.

Metodi

Enable	Nomeoggetto.Enable=True abilita il controllo del bit Nomeoggetto.Enable=False disabilita il controllo del Bit

Eventi

Nessuno

Esempio su un asse rotativo continuo

Variabili Globali

MasterEnc	Long	Encoder Master dell' asse rotativo
VirtualEnc	Long	Encoder virtuale da 0 a Impulsimaster
ImpulsiMaster	Long	Impulsi Encoder Master
PulsBit	Bit	Variabile Bit

Proprietà oggetto CammaBitP1

Fase=300
Durata=20
Master=VirtualEnc
VarBit=PulseBit

'Init TASK PLC

ImpulsiMaster=1024 **'Impulsi encoder master**

Funzione calcolo Modulo nelle funzioni MAIN

```
'Calcola Il modulo
function  Modulo(V as long, M as long) as long
        Dim Ris as long
        Dim P as float
        Dim A as Long

        P =V/M
        A =P                            prende parte intera
        P = P - A             ' prende il resto della divisione
        Ris = P * M                     ' prende il modulo
        if Ris < 0
                Ris =Ris + M        ' Porta sempre in positivo
        endif
        Modulo= Ris
endfunction

'Task Plc
VirtualEnc=CalcoloModulo(MasterEnc,ImpulsiMaster)
'azzera fatto
if CammaBitP1.Fatto=1 && VirtualEnc< CammaBitP1.Fase
        CammaBitP1.Fatto=0
endif
```

3.4 CfiltroVol.vco – Filtro per VOLANTINI ELTTRONCI o ENCODERS

Contiene oggetti che filtrano una variabile, dando in uscita un incremento medio.
Il suo utilizzo è associato a valori che vengono letti da encoder esterni e successivamente utilizzati come movimentazione assi es:

VOLANTINI ELETTRONICI
ENCODER MASTER PER CAMME

Non avendo questi un sincronismo con il sistema si può verificare un effetto di RUMOROSITA' sull' asse SLAVE. Per evitare questo CfiltroVol effettua una media sui VALORI letti.

Hardware ***Tutti***

Proprietà principali

N. elementi	Numero di elementi con cui e' composto il filtraggio. Più alto è il numero, maggiore è l' effetto del filtro, ma più lento è la risposta del sistema. Valori = 10 danno un buon compromesso. Non gestibile in run time
Molt. Filtro	Valore di del moltiplicatore degli elementi calcolati. Se uguale ad 1 , gli impulsi della SORGENTE MASTER vengono trasferiti al motore in modo diretto. Non gestibile in run time ma può essere cambiato tramite il metodo Moltiplica
Encoder	Variabile Sorgente da filtrare. Non gestibile in run time
Variabile	Variabile Destinazione del contenuto del filtro. Questa può essere inviata ad PDO, FILTRO PID ecc. Non gestibile in run time

Metodi

Enable	Nomeoggetto.Enable=True abilita il filtro Nomeoggetto.Enable=False disabilita il filtro
Moltiplica	Nomeoggetto.Moltiplica = valore Cambia il valore di moltiplicazione del volantino.

Eventi

Nessuno

3.5 CInterpPos.vco

OBSOLETO

3.6 MonoAx.vco – Posizionatore MONOASSE completo

Monax è un oggetto che si occupa della gestione completa di un ASSE sia questo CanOpen, +-10V con retroazione da encoder, STEP IDR, ETHERCAT ecc.
Questo racchiude tutte le proprietà, Metodi ed eventi per la gestione completa di un posizionatore evoluto
in pratica un SISTEMA MONOASSE COMPLETO di RAMPE, FINECORSA, GESTIONE VOLANTINO ESTERNO, POTENZIOMETRO OVERRIDE ecc.
L' oggetto opera tutti i suoi calcoli su UNA VARIABILE che successivamente viene associata all' asse. (PDO se CanOpen, Filtro PID ecc.)

Hardware ***Tutti***

Proprietà principali

abvol	1 abilita il controllo tramite volantino elettronico esterno
volantino	variabile encoder volantino. non gestibile in run time
uscita	Variabile di uscita posizione teorica (filtro pid o pdo canopen) non gestibile in run time
Vel	Velocità di spostamento asse La velocità dipende dal campionamento impostato. In pratica sono gli impulsi a campionamento
Vmax	Valore massimo della velocità dell' asse in impulsi
Acc	Accelerazione asse espressa in impulsi di incremento velocità campionamento
Dec	Decelerazione asse espressa in impulsi di Decremento velocità a campionamento
Abs	Spostamenti assoluti o relativi 1 Gli spostamenti sono considerati assoluti dalla quota di 0 0 Gli spostamenti sono relativi dalla quota attuale
Vper	Valore percentuale della velocità. (riferita a maxvper) Questo valore può essere ad esempio controllato da un potenziometro per effettuare un limitazione della velocità.
MaxVper	Valore massimo della VPER
LimitSwP	Valore in impulsi del limite software positivo (per disabilitare usare LIMTION)
LimitSwN	Valore in impulsi del limite software negativo (per disabilitare usare LIMTION)
LimitHwP	Variabile BIT che contiene l' ingresso dell' extra corsa positivo (non gestibile in run time)
LimitHwN	Variabile BIT che contiene l' ingresso dell' extra corsa negativo (non gestibile in run time)
MolVol	Moltiplicatore impulsi volantino elettronico
Nelem	Numero elementi per filtro su volantino elettronico (impostabile i base al tipo di asse)
QuickDec	Valore di decelerazione per quick stop (intercettazione limiti Hw e Sw)
LimitOn	Abilitazione limiti gestita a bit ***Bit 0 Limiti Hw attivi*** ***Bit 1 Limiti Sw attivi*** ***Bit 2 Limiti su volantino elettronico abilitati (abilita i limiti bit0 e bit1)*** ***Bit 3 Limiti su spostamenti abilitati (abilita i limiti bit0 e bit1)***
Fczero	Variabile BIT di fine corsa riferimento HOMING (non gestibile in run time)
Vzero	Velocitá alta per ricerca Fczero (quota negativa fino ad intercettazione fc)
Vfine	Velocita' bassa per posizionamento su fczero in modo preciso
Senso	Senso rotazione asse Questa è una variabile che indica il senso di rotazione dell' asse 1 Senso orario 0 Senso antiorario

VARIABILI GLOBALI

STATUS WORD (gestita a bit read only)

.status	***Bit 0***	***Controllo abilitato***
	Bit 1	***Movimento in corso***
	Bit 2	***Volantino elettronico abilitato***
	Bit 3	***0 Direzione movimento in negativo – 1 Direzione movimento in positivo***
	Bit 4	***Limite HW positivo raggiunto***
	Bit 5	***Limite HW negativo raggiunto***
	Bit 6	***Limite SW positivo raggiunto***
	Bit 7	***Limite SW negativo raggiunto***
	Bit 8	***Homing in corso***
	Bit 9	***0 spostamenti relativi – 1 spostamenti assoluti***
.Post	Posizione teorica asse attuale (read only)	
. _vela	***V***elocita' attuale asse (read only)	

Metodi

.enable	1 abilita il controllo 0 disabilita
.Start	1 start movimento a quota target ***.Quota*** 0 stop movimento in corso
.StartHome	1 Start ricerca homing Sequenza ricerca: ***1 Spostamento in negativo a VZERO fino ad FCzero*** ***2 Spostamento in positivo a VFINE fino a rilascio FCzero*** ***3 Spostamento in negativo a VFINE fino a FCzero*** 0 Stop ricerca homing
.Home	Valore di preset quota es: .Home=1000 preset asse a 1000 count

Eventi

OnEndMove	Viene richiamato a fine movimento

Esempio Ricerca Homing

Variabili Globali

StatoRicerca long

```
'Init Main
MonoAx1.enable=true
MonoAx1.Vzero=1000          ' V alta ricerca sensore
MonoAx1.Vfine=200           ' V fine ricerca sensore
MonoAx1.Acc=20              'Accelerazione
MonoAx1.Dec=20              'Decelerazione


'Start Ricerca inserire il codice nella MAIN
'Start condizionato da Bit (es Ingresso)
If StartHoming=true
        MonoAx1.StartHome=1
        StatoRicerca=10
Endif

CicloZero()         'Chiama la funzione di controllo ciclo di zero

'Codice inserito nelle funzioni MAIN
Function CicloZero() As Void
```

```
        if StatoRicerca=0
                return
        endif
        Select StatoRicerca
                case 10
                        'Controlla status Word homing finito
                        if  MonoAx1.status & 0x100          ' Homing in corso
                                return
                        else
                                StatoRicerca=20
                        endif
                case 20
                        ' Preset asse a quota 0
                        MonoAx1.Home=0
                        StatoRicerca=0              ' Fine Homing
        EndSelect
EndFunction
```

Esempio Motion 4 posizioni con set uscita fine movimento e ritardo 100 Ms

Variabili Globali

```
VectPos(4) Long , VectVel(4) Long , PuntPos Long , StatoCiclo long
'Init Main
MonoAx1.enable=true
MonoAx1.Acc=20                  'Accelerazione
MonoAx1.Dec=20                  'Decelerazione
MonoAx1.Abs=1           'movimenti assoluti
'vettore posizioni
VectPos(0)=3000
VectPos(1)=5000
VectPos(2)=8000
VectPos(3)=3500
'vettore velocita'
VectVel(0)=6000
VectVel(1)=6000
VectVel(2)=3000
VectVel(3)=8000
Start Motion inserire il codice nella MAIN condizionato da Bit (es Ingresso)
If StartMotion=true
        StatoCiclo=10
Endif
CicloMotion()     'Chiama la funzione di controllo ciclo
'Codice inserito nelle funzioni MAIN
Function CicloMotion() As Void
        if StatoCiclo=0
                return
        endif
        Select  StatoCiclo
                case 10
                        MonoAx1.Quota=VectPos(PuntPos)
                        MonoAx1.Vel=VectVel(PuntPos)
                        MonoAx1.Start=true
                        StatoCiclo=20
                case 20
                        'Controlla status Word movimento finito
                        if  MonoAx1.status & 2    ' Movimento in corso
                                return
```

```
                    else
                        OUT1=true                ' Abilita Uscita
                        Tempo=100/TAU ' Delay 100 Ms
                        StatoCiclo=30
                    endif
                case 30
                    if Tempo=0
                        OUT1=false        ' Disabilita Uscita
                        if  PuntPos=0      ' Ciclo finito
                            MonoAx1.Quota=0 ' Riporta asse a 0
                            MonoAx1.Vel=10000
                            MonoAx1.Start=true
                            StatoCiclo=0
                        else
                            Inc PuntPos        'Incrementa posizione
                            StatoCiclo=10
                        endif
                    endif
        EndSelect
EndFunction
'Codice inserito nel task PLC
if Tempo>0
        Dec Tempo ' Decrementa Timer
endif
```

Esempio Gestione Potenziometro di Velocità su scheda NGM EVO Canale Analogico 1
Proprietà da progetto
MonoAx1.volantino Encoder

```
'Init Main
MonoAx1.enable=true
MonoAx1.MaxVper=4096 ' 12 Bit per ingresso analogico NGM EVO

'Codice inserito nel task PLC
MonoAx1.Vper=Ng_Adc(0) ' Lettura canale Analogico 0
```

Esempio Gestione Volantino preso da variabile Encoder NG35 NGIO canale 1
Variabili Globali
Encoder Long

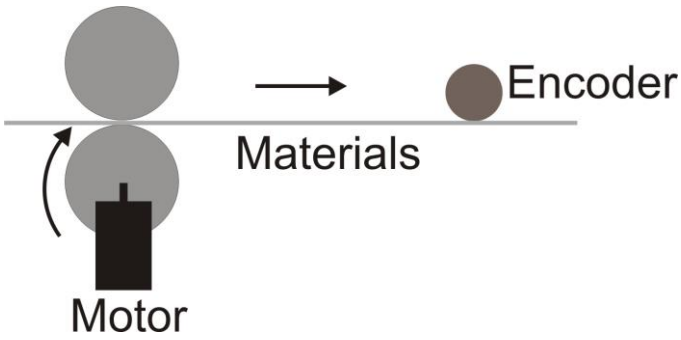
```
'Init Main
MonoAx1.enable=true
MonoAx1.AbVol=1         ' Abilita lettura volantino
MonoAx1.MolVol=1        ' Moltiplicatore ad 1
MonoAx1.Nelem=10        ' 10 elementi di filtraggio per togliere rugosita'

'Codice inserito nel task PLC
Ng_Enc(0,Encoder())     ' Legge encoder da canale 1
```

3.7 MonoAxEnc.vco – Posizionatore con gestione ENCODER per slittamento materiale

Questa classe gestisce un posizionatore completo che a differenza di MonoAx, effettua un POSIZIONAMENTO PRECISO su un ENCODER ESTERNO. Generalmente questo viene utilizzato quando si hanno problemi di SLITTAMENTO DEL MAERIALE (es Taglio di lamiere).
La posizione viene gestita da un ENCODER in modo da recuperare eventuali perdite

Hardware ***Tutti***



L' oggetto gestisce il posizionamento su un Motore (Brushless ecc.) tramite l' encoder viene recuperato eventuali slittamenti.

Proprietà principali

Enable	**True** abilita il controllo dell' asse principale (Motore) **False** Disabilita
Uscita	variabile uscita della quota MOTORE (PDO, PID ecc.). non gestibile in run time
Vel	Velocità di spostamento asse La velocità dipende dal campionamento impostato. In pratica sono gli impulsi a campionamento
Vmax	Valore massimo della velocità dell' asse in impulsi

Acc Accelerazione asse espressa in impulsi di incremento velocità campionamento
Dec Decelerazione asse espressa in impulsi di Decremento velocità a campionamento
Abs Spostamenti assoluti o relativi
1 Gli spostamenti sono considerati assoluti dalla quota di 0
0 Gli spostamenti sono relativi dalla quota attuale
Vper Valore percentuale della velocità. (riferita a maxvper)
Questo valore può essere ad esempio controllato da un potenziometro per effettuare un limitazione della velocità.
MaxVper Valore massimo della VPER
QuickDec Valore di decelerazione per quick stop (intercettazione limiti Hw e Sw)
Senso Senso rotazione asse
Questa è una variabile che indica il senso di rotazione dell' asse
1 Senso orario
0 Senso antiorario
Posr Variabile encoder esterno (questa deve essere convertita in unità di misura del motore)
Spazio Frenatura Anticpo per frenatura del motore prima di arrivare a misura
Questo valore anticipa la posizione di target impostata.
Il motore viene fermato a questo valore e successivamente viene fatto un posizionamento preciso tramite encoder esterno a velocità calcolata dal sistema

Metodi

Start Start posizionamento a quota impostata
True fa partire il posizionamento
False stop asse
Quota Imposta la quota d target del motore
Home Imposta la posizione di ZERO del motore
0 Motore in posizione di 0

VARIABILI GLOBALI

STATUS WORD (gestita a bit read only)

.status	***Bit 0***	***Controllo abilitato***
	Bit 1	***Movimento in corso***
	Bit 2	***Non utilizzato***
	Bit 3	***0 Direzione movimento in negativo – 1 Direzione movimento in positivo***
	Bit 4	***Non utilizzato***
	Bit 5	***Non utilizzato***
	Bit 6	***Non utilizzato***
	Bit 7	***Non utilizzato***
	Bit 8	***Non utilizzato***
	Bit 9	***Assoluto relativo***

Eventi

OnEndMove Viene richiamato a fine posizionamento

3.8 CobjInterpola.vco – Interpolatore fino a 9 assi MULTIPROCESSO

La classe CObjInterpola è la base per interpolazione ASSI. Può essere inserita su qualsiasi tipo di hardware e il numero di assi da interpolare è configurabile come è configurabile anche la tipologia degli assi:

CanOpen
+/-10V
Step/Dir
Ethercat

Questi possono anche essere combinati insieme tra di loro.
In generale le varie funzioni di movimentazione dell'interpolatore lavorano su di un vettore di quote di n posizioni, dove n è il numeri di assi da gestire. Prima di ogni operazione sarà quindi necessario inserire all'interno di questo vettore, per ognuno degli assi gestiti, i valori delle quote target. Anche il numero dei tratti del buffer di movimentazione (**lookahead**).
Ovviamente la programmabilità del n° di assi e di tratti, trova un limite nella disponibilità di memoria del sistema, quindi si dovrà operare una giusta proporzione fra queste caratteristiche.

Nello stesso sistema HARDWARE possono quindi convivere più OGGETTI INTERPOLATORI che gestiscono assi indipendenti ottenendo in questo modo INTERPOLAZIONI MULTIPROCESSO.

Per il dettaglio dei vari metodi e proprietà si rimanda al manuale GUIDA ALL' USO DI VTB capitolo:

FUNZIONI DI INTERPOLAZIONE ASSI

Hardware ***Tutti***

Proprietà principali

N.assi	Numero degli assi che l'interpolatore deve gestire. - Non gestibile in run time
N.tratti	Numero dei tratti del Buffer di movimentazione - **lookahead** . Non gestibile in run time
Vper	Variabile interna valore percentuale della velocità assi. Gestibile in run time
Div.Vper	Numero divisioni della velocità Vper. Valore di default 4096. Non gestibile in run time
Abilita Arcto	Abilitazione utilizzo funzione arcto. Non gestibile in run time
ACC	Accelerazione interpolatore. Utilizzata sia in start sia in stop.
SGLP	Angolo limite per fermata su spigolo del tratto
PC()	Vettore che contiene le quote degli assi gestiti. Ha lunghezza **n**, con n numero assi gestiti.

Metodi

Stop	Stop Assi con Accelerazione programmata ed attesa assi fermi
Fstop	Stop Assi con Accelerazione programmata **SENZA** attesa assi fermi
Move	Restituisce lo stato di movimento dell'interpolatore
Moveto	Interpolazione lineare su tutti gli assi.
Lineto	Interpolazione lineare sul piano di lavoro
Arcto	Interpolazione circolare sul piano di lavoro
Preset	Preset quoat assi
Setpiano	Setta il piano di lavoro.

Eventi

Nessuno

Di seguito vengono riportati alcuni esempi

Regolazione velocità "al volo"

Tramite le due proprietà ***Vper*** e ***Div.Vper*** è possibile modificare al volo, ossia durante l'esecuzione dei vari tratti, la velocità di interpolazione. In pratica la velocità che viene impostata per ogni movimento, viene poi ad ogni campionamento moltiplicata per il rapporto ***Vper/Div.Vper***. Quindi se ad esempio si vuole controllare la velocità di movimento con la lettura di un potenziometro esterno, considerando che il valore max dell'ADC della NGM EVO è di 4096, basterà inserire come ***Div.Vper*** appunto 4096 e come ***Vper*** assegnare la variabile in cui viene letto il valore del potenziometro.

Proprietà oggetto interpolatore es: su scheda NGM EVO

Vper = ngadc (variabile interna ngadc)
Div.Vper = 4096 (canale analogico NGM EVO a 12 BIT)

Task PLC

```
Ngadc=ng_adc(0)  'legge l'ingresso analogico 1 della NGM EVO
```

Automaticamente l'interpolatore provvederà a variare la velocità di movimento al variare dell'ingresso analogico (potenziometro). Agisce su tutte le funzioni di movimentazione, MoveTo, LineTo, ArcTo.

Preset assi

Nomeoggetto.preset(vect_pos())

Questo metodo, da la possibilità di settare la posizione attuale degli assi su una qualunque quota. Può per esempio risultare utile nel caso di una ricerca di zero da effettuare su di un fine corsa.
A completamento della procedura, con gli assi fermi nella posizione voluta

```
pos_vect(0)=0    'asse X
pos_vect(1)=0    'asse Y
pos_vect(2)=0    'asse Z
pos_vect(3)=0    'asse A
pos_vect(4)=0    'asse B
pos_vect(5)=0    'asse C
nomeoggetto.preset(pos_vect())
```

(supponendo di avere un sistema a 6 assi), l'interpolatore azzererà la quota attuale di tutti gli assi, nel punto attuale.

Setpiano

Nomeoggetto.setpiano(ax1,ax2)

Setta il piano di lavoro per LineTo e ArcTo sugli assi indicati. Di default il piano è settato su X,Y (ax1=0 ax2=1).
ax1 non può essere uguale ad ax2. Gli assi assumono la seguente numerazione:

0	X
1	Y
2	Z

Etc…

MoveTo

Nomeoggetto.moveto(vel,ferma,vect_pos)

Movimentazione con interpolazione lineare degli assi indicati. L' interpolazione viene eseguita alla velocità vel su tutti gli assi gestiti. Quindi la velocità è calcolata su tutti gli assi in movimento. Il parametro ferma definisce se gli assi devono fermarsi nel punto finale o continuare al movimento successivo.
Questo prevede che ci siano più movimenti nel buffer dei movimenti.

Parametri

vel	velocità interpolazione
ferma	fermata al termine del tratto
vect_pos	vettore quote finali assi

Parametri di ritorno

Char	0 Non inserito nel buffer (buffer movimenti pieno)
	1 Tratto inserito nel buffer

LineTo

Nomeoggetto.lineto(vel, vect_pos)

Movimentazione con interpolazione lineare degli assi selezionati per il piano di lavoro. Il calcolo della velocità vel di interpolazione viene eseguito sugli assi del piano di lavoro, non viene considerata la velocità di altri eventuali assi contemporaneamente in movimento. Si deve quindi prestare attenzione al caso di movimenti molto brevi sul piano, associati, ad esempio, ad un movimento lungo su di un terzo asse. Questo per andare in posizione contemporaneamente ai due assi del piano, non avendo la sua velocità considerata nel calcolo globale, tenderà a posizionarsi con uno "scatto", che non sempre risulterà eseguibile.
La fermata degli assi viene controllata in base alla proprietà **SGLP** sui due assi del piano di lavoro settato. Se lo spigolo formato da due tratti è maggiore di **SGLP** viene comunque effettuata una fermata.
Questo prevede che ci siano più movimenti nel buffer dei movimenti.

Parametri

vel	velocità interpolazione
vect_pos	vettore quote finali assi

Parametri di ritorno

Char	0 Non inserito nel buffer (buffer movimenti pieno)
	1 Tratto inserito nel buffer

ArcTo

Nomeoggetto.arcto(vel, tipo, vect_pos,i,j)

Movimentazione con interpolazione circolare sugli assi settati del piano di lavoro, con centro.
Gli assi del piano di lavoro effettuano un interpolazione di tipo circolare, mentre gli altri assi del tipo lineare.
In modo analogo a LineTo, la velocità vel è calcolata sul piano e la proprietà ***SGLP*** identifica la fermata sul tratto successivo.
Il senso dell' interpolazione circolare ORARIA o ANTIORARIA è determinato dal parametro tipo.

Parametri

vel	velocità interpolazione
tipo	tipo di arco da eseguire , 2 orario, 3 antiorario
vect_pos	vettore quote finali assi
i,j	coordinate del centro dell'arco

Parametri di ritorno

Char	0 Non inserito nel buffer (buffer movimenti pieno) 1 Tratto inserito nel buffer >1 Errore Arco Impossibile

Move

Nomeoggetto.move()ì

Ritorna lo stato di movimento in corso.

Parametri

Nessuno

Parametro di ritorno

char	**0** Nessun movimento in corso **1** Movimento in corso

Stop

Nomeoggetto.stop()ì

Stop movimento assi con accelerazione programmata ed attesa assi fermi

Parametri

Nessuno

Parametro di ritorno

Nessuno

FStop

Nomeoggetto.fstop()

Stop movimento assi con accelerazione programmata **SENZA** attesa assi fermi

Parametri

Nessuno

Parametro di ritorno

Nessuno

3.9 CpxCanAx.vco - Controllo assi CanOpen +/-10V su scheda NGQx

Classe riferita al tipo di controllo **CAN AX** di Promax rispondente alle specifiche ***CIA DSP 402***. Il controllo gestisce **2 ASSI** da linea CAN OPEN e li converte nel modo classico in **VELOCITÀ +/- 10 V.** Inoltre può gestire la lettura di **2 ENCODER** incrementali LINE DRIVER, due uscite analogiche +/- 10V, un albero elettrico con un **MASTER** ed UNO SLAVE.

Hardware ***Tutti***

Proprietà principali

Nodo Indirizzo fisico del nodo nella rete CAN OPEN. Non gestibile in run time

AxEl **True** abilita l'asse elettrico
False disabilita l'asse elettrico

Kem Parametro moltiplicativo ASSE ELETTRICO ***(imp slave=Imp Master*KEM/KED***)

Ked Parametro divisivo ASSE ELETTRICO

TRampa Tempo in millisecondi per aggancio albero elettrico. Es. partendo con asse slave fermo e master in movimento, lo slave effettua una rampa di Trampa millisecondi per arrivare alla velocita' richiesta

VMaxX Valore numerico della velocità massima del ASSE X. La velocità viene riferita come unità di misura al FACTOR GROUP delle specifiche DSP 402.

VMaxY Valore numerico della velocità massima del ASSE Y. La velocità viene riferita come unità di misura al FACTOR GROUP delle specifiche DSP 402.

VelX Valore numerico della velocità di spostamento ASSE X. La velocità viene riferita come unità di misura al FACTOR GROUP delle specifiche DSP 402.

VelY Valore numerico della velocità di spostamento ASSE Y. La velocità viene riferita come unità di misura al FACTOR GROUP delle specifiche DSP 402.

AccX Valore numerico dell'accelerazione del asse X, essa è espressa come millesecondi necessari per giungere alla velocità massima partendo da zero.

DecX Valore numerico della decellerazione del asse X, essa è espressa come millesecondi necessari per giungere a zero partendo dalla velocità massima.

AccY Valore numerico dell'accelerazione del asse Y, essa è espressa come millesecondi necessari per giungere alla velocità massima partendo da zero.

DecY Valore numerico della decelerazione del asse Y, essa è espressa come millesecondi necessari per giungere a zero partendo dalla velocità massima.

KpX Costante del errore proporzionale del asse X, l'errore di posizionamento viene moltiplicato per questa costante e il risultato usato per correggere la posizione.

KpY Costante del errore proporzionale del asse Y, l'errore di posizionamento viene moltiplicato per questa costante e il risultato usato per correggere la posizione.

QuotaX Valore numerico della quota di target ASSE X di spostamento. La quota viene riferita come unità di misura al FACTOR GROUP delle specifiche DSP 402.

QuotaY Valore numerico della quota di target ASSE Y di spostamento. La quota viene riferita come unità di misura al FACTOR GROUP delle specifiche DSP 402.

AbsX **True** spostamenti in valore assoluto ASSE X
False spostamenti in valore relativo dal punto attuale ASSE X

AbsY **True** spostamenti in valore assoluto ASSE Y
False spostamenti in valore relativo dal punto attuale ASSE Y

StatoX Stato dell' asse X. Vedi stato asse

StatoY Stato dell' asse Y. Vedi stato asse

Metodi

EnableX **True** abilita il controllo del motore ASSE X
False disabilita il controllo del motore ASSE X

EnableY **True** abilita il controllo del motore ASSE Y
False disabilita il controllo del motore ASSE Y

StartX **True** start posizionamento in velocità o posizione ASSE X
False stop movimentazione in corso ASSE X

StartY **True** start posizionamento in velocità o posizione ASSE Y

	False stop movimentazione in corso ASSE Y
HomeX	**True** definisce la posizione di home ASSE X
HomeY	**True** definisce la posizione di home ASSEX
PosrX	Riporta la posizione reale dell' asse X
PostX	Riporta la posizione teorica dell' asse X
PoseX	Riporta l' errore di inseguimento dell' asse X
PosrY	Riporta la posizione reale dell' asse Y
PostY	Riporta la posizione teorica dell' asse Y
PoseY	Riporta l' errore di inseguimento dell' asse Y
ServoX	Soglia errore di inseguimento ASSE X genera evento OnErrorX
ServoY	Soglia errore di inseguimento ASSE Y genera evento OnErrorY
OuotaX	Scrive nell' uscita analogica ASSE X I valori devono essere comprese tra -2048 e +2047 (-10V +10V) **NON DEVE ESSERE ABILIATO IL PID ENABLEX=FALSE**
OutaY	Scrive nell' uscita analogica ASSE Y I valori devono essere comprese tra -2048 e +2047 (-10V +10V) **NON DEVE ESSERE ABILIATO IL PID ENABLEY=FALSE**
InpA	Legge il valore dell' ingresso analogico a 10 bit

Eventi

OnEndMoveX	Si verifica alla fine del movimento ASSE X,
OnEndMoveY	Si verifica alla fine del movimento ASSE Y
OnErrorX	Si verifica quando avviene un errore relativo al controllo asse X
OnErrorY	Si verifica quando avviene un errore relativo al controllo asse Y

RIFERIMENTO DI HOME

Utilizzando il metodo NomeOggetto.HomeX/Y = 1 si definisce la posizione attuale come posizione di ZERO

ALBERO ELETTRICO

L' albero elettrico presente nella scheda CAN AX è di utilizzo semplice ed immediato. In pratica lo SLAVE viene connesso nell' ASSE X, mentre il master nell' asse Y. Il rapporto con cui lo slave segue il master viene dato dai parametri .KEM e .KED, dove KEM è il parametro moltiplicativo degli impulsi master e KED è il parametro divisivo degli impulsi master.

IMP. SLAVE = IMP MASTER * KEM / KED

KEM = 1 e KED =1 un impulso slave corrisponde ad un impulso master - rapporto 1:1

KEM = 2 e KED =1 due impulsi slave corrispondono ad un impulso master - rapporto 2:1

KEM = 1 e KED =2 un impulso slave corrisponde ad due impulsi master - rapporto 1:2

Per abilitare l' asse elettrico è sufficiente inserire il parametro **.AxEl = True,** viceversa per disabilitarlo a FALSE.
Inoltre viene gestito anche il tempo di aggancio **Trampa**. Questo risulta fondamentale quando lo slave deve essere agganciato ad un master in movimento. Non utilizzando Trampa, lo slave avrebbe un Accelerazione proporzionale alla velocità del MASTER e al rapporto di inseguimento. Questa accelerazione potrebbe essere talmente elevata da non essere sopportata dal sistema.
Inserendo un valore di **Trampa**, viene effettivamente generata una Rampa sull' asse slave, trascorso questo tempo, l' asse SLAVE risulta agganciato in **ALBERO ELETTRICO al MASTER**.

STATO ASSE
Lo stato dell' asse X e Y viene controllato tramite la proprietà STATOX e STATOY. Questa e' un byte nel quale ogni bit segnala una determinata condizione:

Nr. Bit	Sigla	Read/Write	Descrizione
0	**ACK**	**R/W**	Se utilizzato viene resettato da applicazione prima di ogni comando da CAN. Quando il suo stato ritorna ad 1, la scheda CAN AX ha accettato il comando. Es. se viene passato un comando di movimento, **resettare ACK**, prima di testare la fine movimento occorre che **ACK** si stato settato, altrimenti il comando non e' stato ancora processato.
1	**MOVE**	**R**	Se settato asse in movimento,altrimenti asse fermo
2	**ERROR**	**R**	Se settato asse in errore.
3	**ENABLE**	**R**	Se settato il relativo asse e' abilitato al controllo di spazio
4	**AXEL**	**R**	Se settato, l' asse è in albero elettrico
5	**SYNC**	**R**	Viene settato quando e' stato raggiunto il sincronismo perfetto dell' albero elettrico, cioe' quando è trascorso il tempo **TRampa**.
6	**RESERVED**		
7	**RESERVED**		

3.10 CstdCanOpen.vco – Gestione DRIVES CanOpen DS301 DS402

Classe riferita ai driver che rispondono alle specifiche CIA DS 402 o DS301.
La libreria contiene marche di Driver commerciali. Nel caso in cui la marca non sia presente, viene riportato un OGGETTO STANDARD DS402.

Hardware ***Tutti***

Proprietà principali

Nodo Indirizzo fisico del nodo nella rete CAN OPEN. Non gestibile in run time

Modo Modo di funzionamento del dispositivo.
0 = Position Mode
1 = Velocity Mode
2 = Interpolation Mode

Velocita Valore numerico della velocità di spostamento. La velocità viene riferita come unità di misura al FACTOR GROUP delle specifiche DSP 402.

Quota Valore numerico della quota di target di spostamento. La quota viene riferita come unità di misura al FACTOR GROUP delle specifiche DSP 402.

Abs **True** spostamenti in valore assoluto
False spostamenti in valore relativo dal punto attuale

EnStato **True** abilita gli eventi **OnEndMove** e **OnError**
False disabilita gli eventi **OnEndMove** e **OnError**
ATTENZIONE
Con EnStato=true viene abilitato l' invio di OGGETTI SDO sulla linea
Questo potrebbe causare un rallentamento dell' applicazione.
Quindi si consiglia di abilitare EnStato solo quando necessario

Metodi

Enable	**True** abilita il controllo del motore
	False disabilita il controllo del motore
Start	**True** start posizionamento in velocità o posizione
	False stop movimentazione in corso
Home	Valore della posizione di home
Posr	Riporta la posizione reale dell' asse
Post	Riporta la posizione teorica dell' asse
PosE	Riporta l' errore di inseguimento dell' asse

Eventi

OnEndMove	Si verifica alla fine del movimento, cioè quando è avvenuto un posizionamento alla quota di target impostata se abilitato **EnStato**
OnError([Coderr as Long)	Si verifica quando avviene un errore relativo al controllo assi Codice di errore dell' evento On Error se abilitato **EnStato**

RIFERIMENTO DI HOME

Utilizzando il metodo NomeOggetto.Home = Valore si definisce la posizione di Home con il valore assegnato. Valore può essere qualsiasi variabile di VTB. Quindi NomeOggetto.Home=0 definisce come punto ZERO la posizione attuale dell' asse, NomeOggetto.Home=1000 definisce che la posizione dell' asse nel punto attuale ha valore 1000.
Nell' esempio riportato di seguito viene rappresentato la gestione di un oggetto motor rappresentando in modo significativo la semplicità di gestione. Il semplice progetto e composto da più oggetti contenuti in una pagina, per chiarezza viene rappresentato l' oggetto e il relativo codice associato all' evento. Ovviamente deve essere presente anche l' oggetto motor che in questo caso ha come nome motor1.

```
'Position Mode
motor1.modo=0

'Abilitazione/Disabilitazione motore
motor1.Enable=True          ' Abilita il controllo del motore
motor1.Enable=False         ' Disabilita il controllo del motore

'Spostamento ASSOLUTI/RELATIVI
motor1.Abs=True         ' Abilita la movimentazione assoluta
motor1.Abs=False            ' Abilita la movimentazione relativa
'Start motion
motor1.Vel=1000 ' Vel
motor1.acc=200  ' acc
motor1.dec=150  ' dec
motor1.Start=True       ' Start movimentazione alla quota target motor1.Quota

'Stop motion
motor1.Start=False      ' Ferma eventuali movimentazioni in corso
```

SET INTERPOLATION MODE

Settando l' oggetto in Interpolation Mode è possibile collegare a questo gli OGGETTI di MOTION che operano su variabili generiche ex: MONOAX, COBJINTERPOLA ecc.
In questo caso occorre che il Driver in questione sia in grado di utilizzare questo tipo di funzionamento.
Inoltre occorre aver configurato tramite il CONFIGURATORE CANOPEN il PDO della quota interpolata.

Per utilizzare il modo interpolatore i driver deve trovarsi in modo 2 e start=true:

asse.modo=2
asse.start=true

Occorre tenere presente che in questo STATO la piattaforma HARDWARE invia a tempi costanti (definiti dal campionamento del TASK PLC) le quote assi ad ogni singolo DRIVER.
Pertanto onde evitare malfunzionamenti occorre che le quote generate da sistema siano concordi con quelle del driver.

In genere i principali problemi si verificano quando occorre stabilire una posizione di HOME comune alla piattaforma HARDWARE e al DRIVER.
In questa condizione è necessario prima di ogni preset della piattaforma, disabiliare il driver dal modo interpolation, per far si che non consideri le quote inviate dal sistema fino a quando i due valori no sono concordi.

Procedura corretta di Preset asse in interpolation mode:
asse.modo=0 **'driver in position mode**
asse.start=false **' disabilita lettura PDO sul driver**
preset della piattaforma (interpolatore.preset... ecc.)
asse.home=***quota presettata sulla piattaforma***

'a questo punto le quote della piattaforma sono concordi con quelle del driver

asse.start=true **' riabilita lettura PDO sul driver**
asse.modo=2 **'driver in interpolation mode**

3.11 CstdGear.vco – Gestione ALBERI ELETTRICI

Contiene oggetti che definiscono un albero elettrico **SLAVE** che si muove con rapporto impostabile riferito ad un **MASTER**. Il rapporto viene generato da due parametri **KEM** e **KED** i quali si riferiscono ad un **MOLTIPLICATORE** di impulsi e ad un **DIVISORE** di impulsi. In sostanza l' asse slave si muove con la seguente definizione:

Imp Slave = Imp Master * KEM / KED

Tramite i due parametri è possibile selezionare qualsiasi rapporto di trasmissione.

KEM = 1 e KED =1 un impulso slave corrisponde ad un impulso master - rapporto 1:1
KEM = 2 e KED =1 due impulsi slave corrispondono ad un impulso master - rapporto 2:1
KEM = 1 e KED =2 un impulso slave corrisponde ad due impulsi master - rapporto 1:2

L' asse SLAVE può essere inserito o disinserito a piacere dalla funzione di ALBERO ELETTRICO, tramite il metodo **ENABLE**, per fare in modo da funzionare come semplice posizionatore.
Il rapporto asse elettrico può essere cambiato anche ***"Al Volo"***.
Il **MASTER** può essere qualsiasi grandezza numerica proveniente ad esempio da un ENCODER,

L' oggetto **CStdGear** opera su una variabile GENERICA, la quale può a sua volta essere associata ad PDO CanOpen, Ethercat, ad un fiktro PID oppure ad una uscita STE/DIR.

Hardware ***Tutti***

Proprietà principali

KEM	Imposta il valore moltiplicativo dell' albero elettrico
KED	Imposta il valore divisivo dell' albero elettrico
Master	Variabile Sorgente che definisce il MASTER. Non gestibile in run time
Slave	Variabile Destinazione questa può essere associata al tipo di ASSE IN GESTIONE Non gestibile in run time

Metodi

Enable	True abilita il controllo del motore slave in ALBERO ELETTRICO False disabilita il controllo del motore

Eventi

Nessuno

Esempio Gestione Albero elettrico preso da variabile Encoder NG35 NGIO canale 1 e KEM,KED impostabile da Input Digitali
Variabili Globali
Encoder Long
GenericOut Long
Proprietà CStdGear
KEM=1
KED=1
Master=Encoder
Slave=GenericOut

```
'Init Main
CStdGear.enable=true

'task Main
if Input1=true          ' Input 1 imposta rapporto 2:1
        KEM=2
        KED=1
endif
if Input2=true          ' Input 2 imposta rapporto 1:2
        KEM=1
        KED=2
endif
'Codice inserito nel task PLC
Ng_Enc(0,Encoder())     ' Legge encoder da canale 1
```

3.12 CstdStep.vco – Gestione ASSI STEP/DIR su sistemi NGQUARK in CanOpen

La ***CStdStep*** contiene oggetti che si occupano del controllo della movimentazione di assi Passo Passo utilizzando una scheda PROMAX serie NGQUARK (CANIO nelle vecchie versioni) collegata al Master per mezzo del protocollo CAN OPEN.

Hardware ***Tutti***

Proprietà principali

Nodo	Nodo CanOpen della scheda NGQUARK. Non gestibile in run time
Quota	Valore numerico della quota di target di posizionamento.
Stato	**True** abilita gli eventi OnEndMove **False** disabilita gli eventi OnEndMove **ATTENZIONE** **Abilitando questa proprietà, il sistema genera ciclicamente SDO che potrebbero in alcuni casi rallentare il ciclo dell' applicazione. Si consiglia di ABILITARLA solo quando è in corso un posizionamento**

Metodi

I metodi con postfisso 'A' fanno riferimento al primo canale della scheda, 'B' al secondo canale, 'C' al terzo e 'D' al quarto.

AbsA(B-C-D)	**True** abilita il posizionamento assoluto del motore **False** abilita il posizionamento relativo del motore
StartA(B-C-D)	**True** start posizionamento alla quota inserita **False** stop movimentazione in corso
QuotaaA(B-C-D)	Quota di target posizionatore
PosaA(B-C-D)	Quota teorica attuale del motore, il valore è di tipo long
HomeA(B-C-D)	Homex=0 definisce la posizione di home dell'asse
AccA(B-C-D)	Accelerazione. Incrementi Vel (HERTZ) a TAU scheda SLAVE dove TAU =10
DecA(B-C-D)	Decelerazione. Incrementi Vel (HERTZ) a TAU scheda SLAVE dove TAU =10

VelA(B-C-D) Velocità asse in HERTZ

Eventi

OnEndMoveA(B-C-D) Si verifica alla fine del movimento del posizionamento dell'asse, cioè quando è avvenuto un posizionamento alla quota di target

Esempio per due assi

```
Step1.homea=0          ' azzera la quota del canale A
Step1.homeb=0          ' azzera la quota del canale B

Step1.acca=50          ' setta l'accelerazione del canale A
Step1.deca=50          ' setta la decelerazione del canale A
Step1.vela=500         ' setta la velocità del canale A
Step1.quotaa=4000      ' setta la quota del canale A

Step1.accb=100         ' setta l'accelerazione del canale B
Step1.decb=100         ' setta la decelerazione del canale B
Step1.velb=1000        ' setta la velocità del canale B
Step1.quotab=8000      ' setta la quota del canale B

Step1.starta=1         ' muove il motore A alla quota impostata in quotaa
Step1.startb=1         ' muove il motore B alla quota impostata in quotab
```

3.13 CPPpos.vco – Gestione Posizionatore assi STEP/DIR su NGM EVO

Questo oggetto gestisce le uscite STEP/DIR della scheda NGM EVO come posizionatore o velocità. Si possono inserire fino a 4 OGGETTI per coprire tutte e 4 le uscite della scheda NGM EVO.
Rispetto agli oggetti MONOAX che operano come interpolatore, quest' oggetto è in grado di occupare meno risorse e generare frequenze più elevate.

ATTENZIONE
Il canale della scheda NGM EVO non deve essere settato come P-P interpolato (vedi oggetto NGM EVO_init)
Esempio attivazione due CANALI
P-P enable Mask=3
P-P Interp. Mask=0

Hardware ***NGM EVO+... - NGM EVO***

Proprietà principali

NGM EVO Channel Canale scheda NGM EVO (da 0 a 3). Non gestibile in run time
Vel Velocità per movimenti in posizione espressa in HERTZ
Acc Accelerazione espressa in HERTZ a campionamento. Es Campionamento=2 Ms impostando un valore di ACC=10 si passa da 0 1 1000Hz in 200 Msec
Dec Decelerazione espressa in HERTZ

Metodi

MoveRel(Corsa long) Movimento in RELATIVO per un numero di passi indicato in CORSA
MoveAbs(Corsa long) Movimento in ASSOLUTO per un numero di passi indicato in CORSA
MoveVel(Vel long) Movimento in velocità indicata in Vel. La velocità può anche essere cambiata ***"al volo"*** sempre tramite questo metodo
Velocità negativa cambia direzione.
Stop() Stop con Rampa posizionamento o velocità
Move() char Ritorna 0 asse fermo – 1 asse in movimento
Qact() long Ritorna la quota attuale assoluta dell' asse
Preset(quota long) Preset asse a quota assoluta al valore impostato in **quota**

Eventi
Nessuno

Esempio

```
CPPpos1.vel=1000
CPPpos1.acc=10
CPPpos1.dec=10

if inpstart=true    && Move()=0      ' Se ingresso start attivo e asse fermo
        moverel(1000)                ' muove in relativo di 1000 passi
endif
if inpstopt=true                     'se ingresso stop attivo ferma l' asse
        Stop()
endif
```

3.14 CasseMRot.vco - Asse rotativo con cambio velocità in fase

Quest' oggetto gestisce un asse rotativo VIRTUALE MASTER con cambio velocità e STOP in fase all0 interno del GIRO.
Può essere utilizzato per TAGLIO AL VOLO , SALDATURE AL VOLO ecc.
L' asse gira sempre in uno stesso senso e effettua un ricerca di homing iniziale.
L' asse può variare la velocità all' interno del giro per un anglo impostabile.

Hardware ***Tutti***

Proprietà principali

Vel	Velocità in GIRI al minuto
Vel2	Seconda velocità espressa in percentuale rispetto Vel. Questa velocità viene applicata tra **Qstart** e **Qend** ***100*** ***Vel2=Vel*** ***<100*** ***Vel2<Vel*** ***>100*** ***Vel2>Vel***
Velz	Velocità di Homing in Giri/Minuto
Qgiro	Numero di impulsi per GIRO dell' ASSE
Qstart	Quota di inizio cambio velocità da Vel a Vel2. espressa in GRADI (dipende da Qum)
Qend	Quota di fine seconda velocità e ritorno a Vel. espressa in GRADI (dipende da Qum)
Qum	Unità di misura delle quote di default=360 360 Quote in espresse in gradi 3600 Quote in espresse in decimi di grado 36000 Quote in espresse in centesimi di grado ecc.
Acc	Accelerazione espressa in impulsi a campionamento per START partenza asse
Uscita	Variabile associata all' asse reale (PDO ecc.)
Input Zero	Bit sensore di ZERO stato attivo alto

Variabili Globali

Stato Char	Contiene lo stato dell' asse: ***Read Only*** ***0 → Attesa Start*** ***1 → Ciclo normale*** ***2 → Ciclo in fase di STP (attesa fasamento)*** ***3 → Ciclo in STOP***
rzero_ok Char	Contiene lo stato della ricerca di ZERO asse ***Read/Write*** ***0 → Ricerca di zero non effettuata*** ***1 → Ricerca di zero OK*** **La variabile può essere azzerata per eseguire nuovamente la ricerca di zero al prossimo START CICLO**
Qg Long	Contiene la quota TEORICA dell' asse all' interno del GIRO ***Read Only***
Qt Long	Quota TEORICA ASSOLUTA dell' asse ***Read Only***
Sync Char	Viene messo ad 1 quando l' asse passa per lo ZERO ***Read/Write***

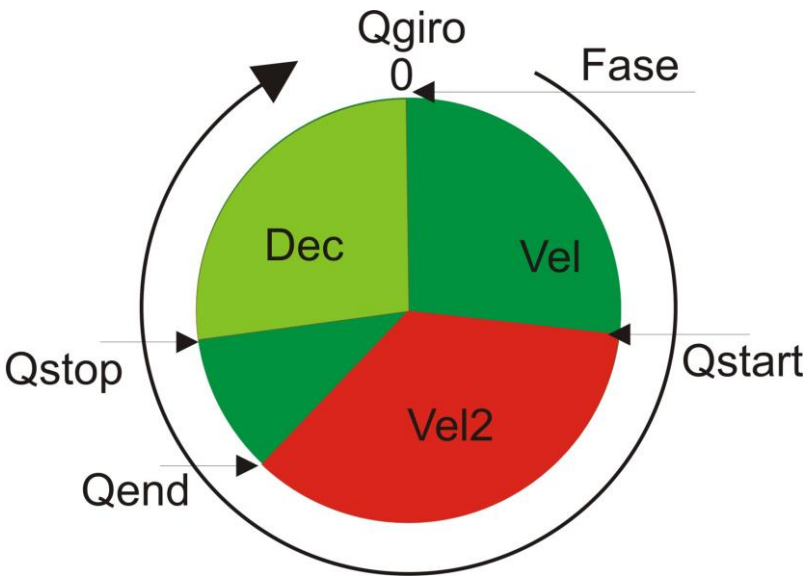
	Viene azzerato da applicazione esterna
Velm Long	Imposta la velocità del ciclo manuale in Giri/Min ***Read/Write***

Metodi

StartI()	Esegue lo START dell' asse. Effettua la ricerca di ZERO ***Rzero_ok=0***
Stop()	Stop con Rampa in fase (l' asse si ferma sullo 0)
Stopi()	Ferma l' asse immediatamente senza Rampa
Startm()	Esegue lo Start dell' asse in ciclo manuale a velocità fissa ***Velm***
Stopm()	Ferma il ciclo manuale con Rampa
Update()	Aggiorna le proprietà in RunTime (Qstart,Qend ecc.) Deve essere chiamata quando si vuole rendere attive le modifiche ai valori effettuate

Eventi

Nessuno



Tratto a Velocità Vel

Tratto a velocità Vel2

Rampa di decelerazione per fermata in fase quando Stop()

3.15 CgenFreq.vco - Generatore di frequenza NGM EVO

Questo oggetto gestisce il generatore di FREQUENZA per le schede NGM EVO.
Si possono inserire fino a 4 Oggetti in modo da gestire i 4 canali della scheda. Utillizzando il canale come generatore di frequenza, non è possibile gestirlo come ASSE STEP/DIR.
Il relativo canale NON DEVE ESSERE ATTIVATO come INTERPOLATO (vedi oggetto NGM EVO_init)

Es. per attivazione primi due canali
P-P enable Mask=3
P-P Interp Mask=0

Questo oggetto consente la generazione di frequenze fino a 10 Mhz senza aggravare tempo della CPU.
L' oggetto è indicato per il pilotaggio di dispositivi in frequenza oppure per essere collegato a convertitori Frequenza/Tensione.
La frequenza viene generata sull' uscita STEP del canale

Hardware ***NGM EVO+... - NGM EVO***

Proprietà principali

NGM-EVO channel	Associa l' oggetto ad uno dei 4 canali della scheda NGM EVO (valore da 0 a 3)
Enable	**True** attiva l' uscita DIR del canale **False** disattiva l' uscita DIR del canale
Freq	Imposta la frequenza in uscita in HERTZ. (start generatore)
Acc	Imposta il valore di Accelerazione espresso in HERTZ per campionamento: Es. cona 2 Ms impostando un valore 10 Hertz, si passa da 0 a 1000 in 200 Msec. Un valore uguale a 0 disabilita la RAMPA
Dec	Come Acc ma imposta la rampa di decelerazione

Variabili Globali

ActFreq Long	Valore attuale della frequenza in uscita

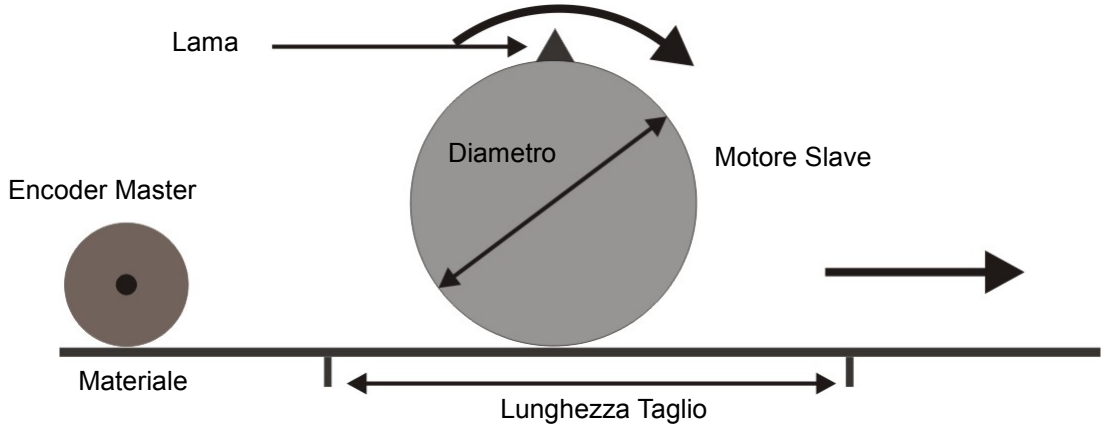
3.16 CTaglioRot.vco – Gestione Taglio "AL VOLO" Rotativo

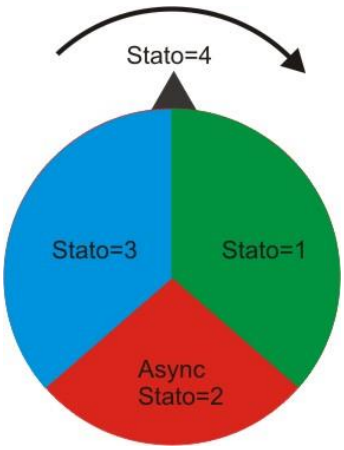
Classe riferita ad oggetti adatti per gestire TAGLOI DI TIPO ROTATIVO. Questi trovano notevoli applicazioni nell' industria es. PINZE PER SALDATURA SACCHETTI DI PLASTICA, TAGLIO DI LAMIERA AL VOLO, TAGLIO DI CARTA AL VOLO ecc.

Hardware ***Tutti***

Principio del taglio rotativo

Il materiale scorre in modo continuo. Una lama o una pinza, effettuano il taglio o saldatura di una determinata lunghezza del materiale, mentre questo è in movimento. La velocità e la misura di spostamento del materiale, vengono rilevati da un encoder (MASTER) generalmente messo a contatto del materiale stesso. Al momento del taglio, la LAMA deve trovarsi in sincronismo perfetto con la velocità del materiale. La fase della rotazione della lama e' determinata dalla lunghezza e dalla velocità del materiale da tagliare.





Proprietà principali

Len	Lunghezza di Materiale da tagliare (L. Taglio) in millimetri
Imaster	Impulsi a giro dell'encoder master.
Sviluppo	Sviluppo in unita' di materiale che passa per un giro master espresso in millesimi di millimetro
Islave	Impulsi che servono allo slave per fare un giro della LAMA
Diam	Diametro della LAMA espresso in millimetri
Async	Angolo in cui la LAMA effettua il TAGLIO con il materiale

	Durante questo angolo l' asse è in sincronismo con il materiale
KSync	Rapporto asse elettrico quando la LAMA TAGLIA il materiale. Questo è espresso in percentuale. 100 significa un rapporto 1:1 tra lama e materiale. Aumentando questo valore, la LAMA aumenta di velocità, diminuendolo, la LAMA diminuisce la velocità
Master	Variabile Sorgente che definisce il MASTER ENCODER.
Slave	Variabile destinazione che definisce SLAVE (PDO, PID ecc.)
Slave Cont.	**False** non viene mai azzerato l' encoder slave. Questo può creare problemi di overflow nella rotazione continua. L' azzeramento in questo caso viene fatto esternamente. **True**, ad ogni giro viene azzerata la posizione dello SLAVE

Metodi

Update	**True** Deve essere utilizzato ogni qualvolta viene effettuato un cambio delle proprietà in run time
Restart	**True** Reinizializza la procedura di taglio azzerando tutti i contatori
Incx	Valore Incremento ad ogni campionamento dello sfasamento impostato
Csfx	Valore Sfasamento tra master e slave espresso in impulsi MASTER
Chiuse	ad 1 indica contatto della lama con il materiale (read only)
Enable	**True** abilita il controllo del TAGLIO **False** disabilita il controllo del TAGLIO
Stato	Stato della LAMA (read only) ***1 prima del sincronismo*** ***2 Angolo di sincronismo*** ***3 Uscita dall' angolo di sincronismo*** ***4 Cambio di 1 giro***

Eventi

Nessuno

3.17 NgmInit.vco – Init scheda NGM EVO

L'oggetto NGM_init, viene aggiunto automaticamente al progetto dal sistema di sviluppo, quando viene selezionato nelle opzioni il codice terminale NGM EVO o NGM EVO/LPCxx.
L'oggetto fornisce una visione completa di tutte quelle che sono le opzioni software da configurare per utilizzare correttamente la NGM EVO.
In particolare permette di impostare:

- L'attivazione del protocollo di comunicazione proprietario PROMAX RPC, con il relativo baudrate
- Quanti e quali sono i canali analogici in ingresso presenti
- Gli assi passo passo da usare e quelli gestiti dall'interpolatore
- Numero schede I/O presenti

Ovviamente, per ogni singolo progetto esisterà un solo oggetto NGM init, visto che uno solo è il CN su cui gira il programma.

Hardware ***NGM EVO+... - NGM EVO***

Proprietà principali

Link RPC port	Porta RS232 su cui abilitare Il protocollo RPC per gestione HOST PC. Valori: **0 Nessun RPC Link** **Porta seriale SER1/PROG** (in questo caso viene disabilitato il DEBUG e il download applicazione deve essere effettuato in modo manuale con i tasti BOOT/RESET della scheda NGM EVO **1 Porta seriale SER2**
Link RPC baud	Baud rate da utilizzare per la comunicazione RPC
ADC enable mask	Maschera di attivazione canali analogici in ingresso. Gestita a bit **Bit 0** canale analogico 0 abilitato (viene escluso input digitale 1) **Bit 1** canale analogico 0 abilitato (viene escluso input digitale 2)

	ecc.
P-P enable mask	Maschera di attivazione assi Passo-Passo **Bit 0** Canale 0 abilitato **Bit 1** Canale 1 abilitato (vengono escluse uscite digitali 9-12) **Bit 2** Canale 2 abilitato (vengono escluse uscite digitali 10-13) **Bit 3** Canale 3 abilitato (vengono escluse uscite digitali 11-14)
P-P Interp. Mask	Maschera di gestione assi Passo-Passo da interpolatore. Gestita a bit **Bit 0** Canale 0 in interpolation mode **Bit 1** Canale 1 in interpolation mode **Bit 2** Canale 2 in interpolation mode **Bit 3** Canale 3 in interpolation mode
Num. NGM-IO	Numero di schede di espansione ingressi/uscite NGMIO presenti. Si ricorda che 16 in e 14 out sono già disponibili nella configurazione std quindi questa non và considerata
L-Sync enable mask	Proprietà riservata per future espansioni
L-Sync Prescaler	Proprietà riservata per future espansioni

Metodi

Nessuno

Eventi

Nessuno

4 CLASSE TEMPORIZZATORI

La Macro classe Temporizzatori contiene oggetti che gestiscono l' utilizzo di temporizzatori. Un temporizzatore genera un evento allo scadere del tempo impostato, di conseguenza viene gestito il relativo codice che gestisce l' evento. Si fa presente che i temporizzatori inseriti nelle TASK MAIN e DI PAGINA non hanno un ciclo perfettamente costante, questo dipende dalla quantità di codice presente nel TASK MAIN o di PAGINA.

Hardware ***Tutti***

4.1 CBitTimer.vco – Timer per gestione bit

Classe che contiene Timer che gestiscono il SET e RESET di variabili bit. Contengono due proprietà che riguardano il tempo di Bit ON e il tempo di Bit Off, cioè il tempo che la variabile bit deve rimanere allo stato logico 1 e il tempo che deve rimanere allo stato logico 0. La variabile Bit può essere di qualsiasi tipo (questa può essere anche una variabile non BIT, in questo caso il valore viene commutato da 1 a 0). Se la variabile Bit è riferita ad un uscita digitale, questa viene settata e resettata nei tempi impostati.

Proprietà principali

Variabile	Variabile bit
BitOn	Tempo in millisecondi che la variabile Bit deve rimanere allo stato logico 1. Proprietà che può essere gestita anche in Run time con Nomeoggetto.BitOn=valore
BitOff	Tempo in millisecondi che la variabile Bit deve rimanere allo stato logico 0. Proprietà che può essere gestita anche in Run time con Nomeoggetto.BitOff=valore
Enable	**True** abilita gli eventi **False** disabilita gli eventi

Metodi

Nessuno

Eventi

OnSet	Si verifica quando la variabile viene settata allo stato logico 1
OnReset	Si verifica quando la variabile viene settata allo stato logico 0

4.2 CStdTimer.vco – Timer generico

Classe che contiene Timer che gestiscono per la gestione di temporizzazioni generiche. Questa genera un evento allo scadere del tempo.

Proprietà principali

Variabile	Variabile bit
Intervallo	Tempo in millisecondi di attivazione TIMER Proprietà che può essere gestita anche in Run time con Nomeoggetto.BitOff=valore
Enable	**True** abilita gli eventi **False** disabilita gli eventi

Metodi

Nessuno

Eventi

OnTimer	Si verifica quando il tempo è scaduto (il timer si riattiva automaticamente)

5 CLASSE COMMMASTER

La Macro classe commmaster contiene la gestione dei più comuni protocolli di trasmissione gestiti da PLC o periferiche esterne. La trasmissione dati viene effettuata in RS232 o RS485 ed è riferita ad un MASTER.

Hardware ***Tutti***

5.1 CommMaster_Modbus.vco – Master Modbus

Classe che la gestione completa di un MASTER modbus RTU

Proprietà principali

BaudRate	baud rate inserire valori concordi con periferiche esterne slave
TimeOut	Tempo in millisecondi del TIME OUT su risposta slave. Questo valore deve essere di norma maggiore al timeout impostato sugli slaves
Parita	0 nessuna - 1 odd - 2 even
N. bit car	Numero bit per carattere
N. bit stop	Numero bit di stop

Metodi

function.write_reg(nodo as char, addr as uint, value as int) as char

Preset single register func 16 ModBus RTU

Parametri

nodo	Nodo slave modbus
addr	Indirizzo registro di partenza scrittura su slave (fare riferimento allo slave)
Value	Unsigned integer contenente i valori da scrivere

Ritorna

0	Scrittura OK
1	Risposta errata dallo slave
2	Time Out
3	lunghezza dati > di 127

Esempio

```
' scrittura su nodo 1 registro 10
' variabile regmodbus
regmodbus=100
Valret=modbusmaster1.write_reg(1, 10, regmodbus)
if valret>0
        ' errore scrittura
endif
```

function.read_reg(nodo as char, addr as uint, value as *int) as char

Read multiple registers func 3 ModBus RTU

Parametri

nodo	Nodo slave modbus
addr	Indirizzo registro di partenza lettura su slave (fare riferimento allo slave)
Value	Puntatore ad unsigned integer di deposito dato letto

Ritorna

0	Scrittura OK
1	Risposta errata dallo slave
2	Time Out
3	lunghezza dati > di 127
4	errore checksum

Esempio

```
' lettura da nodo 1 registro  10 deposito in  regmodbus
Valret=modbusmaster1.read_reg(1, 10, regmodbus(),3)
if valret>0
        ' errore lettura
endif
```

5.2 CommMaster_Omron.vco – Master omron BCD

Classe che la gestione completa di un MASTER con protocollo OMRON BCD

Proprietà principali

BaudRate	baud rate inserire valori concordi con periferiche esterne slave
TimeOut	Tempo in millisecondi del TIME OUT su risposta slave. Questo valore deve essere di norma maggiore al timeout impostato sugli slaves
Parita	0 nessuna - 1 odd - 2 even
N. bit car	Numero bit per carattere
N. bit stop	Numero bit di stop

Metodi

function .write_regn(nodo as char,ad as uint,buf as *uint,n as uint) as char

Scrittura dati nel PLC

Parametri

nodo	Nodo PLC
ad	Indirizzo registro di partenza scrittura
Buf	Puntatore ad unsigned integer contenente i valori da scrivere
n	Numero di registri da scrivere

Ritorna

0	Scrittura OK
1	Errore

Esempio

```
' scrittura su nodo 1 registro di start 10 3 valori
' dati presi da vettore regomron(127) as uint
regomron(0)=100
regomron(1)=200
regomron(3)=300
Valret=omron1.write_regn(1, 10, regomron(),3)
if valret>0
        ' errore scrittura
endif
```

function .read_regn(nodo as char,ad as uint,buf as *uint,n as uint) as char

Lettura dati da PLC

Parametri

nodo	Nodo PLC
ad	Indirizzo registro di partenza lettura
Buf	Puntatore ad unsigned integer valori di deposito letti
n	Numero di registri da leggere

Ritorna

0	Lettura OK
1	Errore

Esempio

```
' Lettura su nodo 1 registro di start 10 3 valori deposito in regomron(127)
Valret=omron1.read_regn(1, 10, regomron(),3)
if valret>0
        ' errore lettura
endif
```

Eventi

nessuno

5.3 TCP_Client.vco – Client TCP/IP

Solo per NG35

Questo oggetto gestisce una comunicazione generica TCP in modo CLIENT e il protocollo RPC per lo scambio dati tra schede PROMAX.

Proprietà principali

IP address	Indirizzo IP remoto a cui connettersi - ***no in Run time***
Port	Porta alla quale connettersi - ***no in Run time***
Idle TimeOut	Time out disconnessione per inattività (in secondi) - ***no in Run time***
RPC TimeOut	Time out risposte protocollo RPC (in millisecondi) - ***no in Run time***
bytes_received	Numero byte nel buffer di ricezione ***– Read Only***
status_connected	Connessione avvenuta ***– Read Only***
status_closed	Connessione chiusa ***– Read Only***
status_abort	Connessione chiusa (da remoto, errore, ecc.) ***– Read Only***
status_overun	Perdita dati in ricezione ***– Read Only***

Metodi

Questi metodi consentono di gestire una comunicazione TCP dal lato CLIENT. Sono indipendenti dal protocollo utilizzato, ovvero i dati in ricezione/trasmissione non hanno un significato definito ma dipendono dal protocollo utilizzato a livello superiore.

function .connect(wait_time *as long*) *as char*

Richiesta di connessione a IP e PORT impostato dalle proprietà. La funzione attende per il tempo impostato (wait_time) che il dispositivo remoto risponda alla richiesta dopo di che sia ha un ritorno dalla funzione.
ATTENZIONE: la gestione della connessione è indipendente da questo tempo. Il parametro “wait_time” serve esclusivamente per uscire dalla funzione ma il sistema continua a riprovare più volte anche se è scaduto questo tempo.
Per capire quando effettivamente il time-out di sistema è scaduto occorre testare la status: almeno un bit tra status_connected, status_closed o status_abort deve essere settato.
Evitare di rieseguire una “connect” quando il time-out di sistema non è ancora scaduto.

Parametri

Wait_time	Tempo di attesa connessione

Ritorna

>0	Connessione OK
-1	Errore di connessione
-2	Time out scaduto

Esempio

```
' Apertura connessione
TCP.connect(0)   ' Non viene utilizzato il time-out della funzione
                 ' ma viene effettuato il test connessione dinamicamente
'……..
if TCP.status_connected
                         'Connessione attiva
' …….
' …….
endif
```

function .close() *as void*

Chiusura connessione. Termina la connessione attiva liberando le risorse di Sistema.

function .send(buf as **char, len as uint*) *as int*

Invio dei dati alla connessione aperta. La funziona effettua l'invio dati e ritorna immediatamente. In caso di errori di rete il sistema effettua vari tentativi dopo di che la connessione viene chiusa.

Parametri

buf	Puntatore ai dati da inviare
len	Numero di Bytes da inviare

Ritorna

>=**0**	Numero di Bytes inviati
-**1**	Errore invio dati

Esempio

```
command(100) as char
bufrx(100) as char
nbyte as int

strcpy(command(),"START") ' Copia START in command
TCP.send(command(),5) ' Invia la stringa

' Attesa risposta
While true
        nbyte=TCP.recv(bufrx(),20) ‘ Attende una risposta di 20 Bytes
        If nbyte>0 ' Processa i dati ricevuti
                .
                exitwhile
        endif
loop
```

function .recv(buf as **char, len as uint*) *as int*

Lettura dati dal buffer di ricezione. Leggendo la proprità **bytes_received** (solo lettura) è possibile sapere quanti dati (bytes) sono presenti nel buffer di ricezione. Il parametro len indica quanti dati scaricare dal buffer. Il valore di ritorno conterrà i dati effettivamente scaricati, sarà normalmente uguale a **len** a meno che non si voglia leggere più dati di quanti siano stati ricevuti. In questo caso ritornerà il valore di dati effettivamente letti.

Parametri

buf	Puntatore ai dati di destinazione
len	Numero Massimo di Bytes da leggere

Ritorna

>=**0**	Numero di dati letti

Esempio

```
command(100) as char
bufrx(100) as char
nbyte as int

strcpy(command(),"START") ' Copia START in command
TCP.send(command(),5) ' Invia la stringa

' Attesa risposta
While true
        nbyte=TCP.recv(bufrx(),20) ‘ Attende una risposta di 20 Bytes
        If nbyte>0 ' Processa i dati ricevuti
                .
                exitwhile
        endif
loop
```

5.3.1 Funzioni protocollo PROMAX RPC

Il protocollo RPC è il sistema di comunicazione presente nelle schede Promax dotate di porta Ethernet. Il server per questo protocollo e sempre alla porta **6000**, occorre quindi avere una connessione aperta a tale porta.

function .rpc_write(ad as *long, len as uint,* buf as **char*) *as int*

Scrittura dati su scheda remota tramite il protocollo RPC.
Il sistema invia una scrittura dati ad un indirizzo di memoria del dispositivo remoto. Per velocizzare il trasferimento dati il sistema non attende che siano scritti effettivamente nel dispositivo remoto. **Se serve un sincronismo occorre effettuare una lettura tramite la funzione .rpc_read magari dei dati stessi.** Questo garantisce un perfetto sincronismo

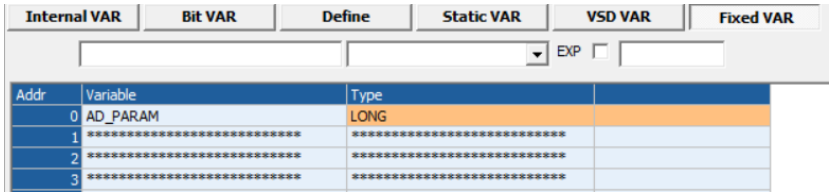
Parametri

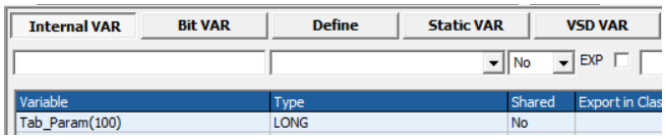
ad	Indirizzo di memoria remoto dove scrivere i dati
len	Numero di Bytes da scrivere
buf	Puntatore ai dati da inviare

Ritorna

>=**0**	Numero di dati scritti
-**1**	Errore di invio
-2	Time out scaduto

Esempio
Nella scheda slave definire **AD_PARAM** nelle fixed (es ad ADDR 0) e un array di nome **Tab_Param** nelle variabili interne:





Assegnare alla Fixed **AD_Param**, l' indirizzo di **tab_param** (fare questo nella Init del TASK PLC)

AD_PARAM=tab_param()

Nella master definire **AD_PARAM** nelle fixed allo stesso indirizzo della slave e definire la tabella parametri della stessa dimensione (nell'esempio 100 char o 50 int o 25 long)

```
' Apre la connessione
TCP.connect(0)
.
.
Tab_param(0)=20
Tab_param(1)=30
.
.
TCP.rpc_read(AD_PARAM(),4, AD_PARAM())          ' lettura puntatore remoto
TCP.rpc_write(AD_PARAM,8,tab_param())           ' scrittura di 8 bytes (2 long)
```

Se si utilizza come slave una NGM-EVO al posto della NG-35 per leggere il puntatore occorre passare manualmente l'indirizzo:

```
TCP.rpc_read(FIXED_EVO+ADDR,4, AD_PARAM())    'lettura puntatore
Dove FIXED_EVO=536874496
```

function .rpc_read(ad as *long, len as uint,* buf as **char*) *as int*

Lettura dati da scheda remota tramite il protocollo RPC.
Il sistema invia una richiesta di lettura dati da un indirizzo di memoria del dispositivo remoto ed attende la risposta. I dati ricevuti vengono scritti tramite il puntatore **buf**.

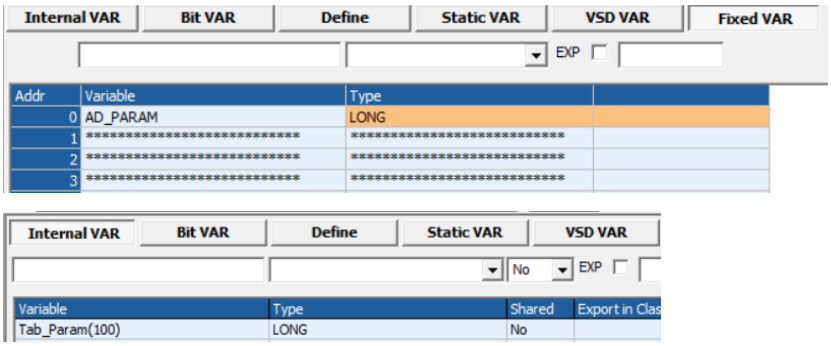
Parametri

ad	Indirizzo di memoria remoto da dove leggere i dati
len	Numero di Bytes da leggere
buf	Puntatore destinazione dei dati

Ritorna

>=**0**	Numero di dati letti
-**1**	Errore di invio
-2	Time out invio
-3	Time out risposta

Esempio
Nella scheda slave definire **AD_PARAM** nelle fixed (es ad ADDR 0) e un array di nome **Tab_Param** nelle variabili interne:



Assegnare alla Fixed **AD_Param**, l' indirizzo di **tab_param** (fare questo nella Init del TASK PLC)

AD_PARAM=tab_param()

Nella master definire **AD_PARAM** nelle **fixed** allo stesso indirizzo della slave e definire la tabella parametri della stessa dimensione (nell'esempio 100 char o 50 int o 25 long)

```
' Apre la connessione
TCP.connect(0)
.
TCP.rpc_read(AD_PARAM(),4, AD_PARAM())        ' lettura puntatore remoto
TCP.rpc_read(AD_PARAM,100,tab_param())        ' lettura di 100 byte
```

6 CLASSE GENERAL

6.1 Cpwm.vco – Gestione uscita PWM su NG-PP

Questo oggetto gestisce l'uscita pwm sulla scheda NG-PP. L'uscita attiva è soltanto quella corrispondente al segnale step del quarto canale (connettore J21).

Proprietà

Chan	Indice del canale sulla scheda NG-PP (attualmente solo 3)
Polarità	Setta la polarità del segnale
Freq	Imposta la frequenza del pwm

Metodi

Name.val(val) as void

Scrive il valore dell'uscita pwm.

Val	Valore del duty cycle da 0 a 1024

Eventi

nessuno

Note

La frequenza è ottenuta mediante un divisore così calcolato:

div = 75000000 / 1024 / Freq

Impostando una frequenza, quella ottenuta è la prima disponibile superiore. Es. impostando 10000 otteniamo 10463Hz.

Queste sono alcune frequenze disponibili:

Div	Freq (Hz)	
2	36621	Freq max
3	24414	
4	18310	
5	14648	
6	12207	
7	10463	
8	9155	
9	8138	
10	7324	
11	6658	
12	6103	
13	5634	
14	5232	
	...	
18	4069	
	...	
24	3052	
	...	
36	2034	
	...	
73	1003	
	...	
36621	2	Freq min

6.2 Cpwm.vco – Gestione uscita PWM su NGM-EVO

Questo oggetto gestisce le uscite pwm sulla scheda NGM-EVO. Ci sono 4 uscite disponibili corrispondenti alle prime 4 uscite digitali. La prima uscita può essere configurata per un uscita analogica.

Proprietà

Enable A bit. Abilita la relativa uscita PWM.
Polarità Setta la polarità del segnale
Center Align Abilita la modalità center align
Freq Imposta la frequenza del pwm
DivisioniNumero di divisioni corrispondenti al 100% (2-65535)

Metodi

pwm_val(id, val) as void
Scrive il valore dell'uscita pwm.
Id Indice del canale (0...3)
Val Valore del duty cycle da 0 a Divisioni

ATTENZIONE: Questa è una funzione di sistema, non deve essere preceduta dal nome dell'oggetto.

Eventi

nessuno

Note

La frequenza massima è data da: 20000000 / Divisioni

6.3 FastInput.vco – Gestione ingressi ad INTERRUPT per NGIO-NGPP-NGMsX-NGQx

Classe per la gestione degli ingressi ad interrupt relativa alle schede NGIO, NGPP, NGMsX, NGQx.
Solo determinati ingressi sono abilitati all' INTERRUPT.
La gestione ad INTERRUPT di un ingresso, permette di leggerlo alla velocità massima gestita dall' hardware (normalmente nell' ordine o sotto i microsecondi).
Queste schede mettono a disposizione i seguenti ingressi ad INTERRUPT:

NGIO	→ Ingresso Tacca di Zero per ogni canale (2 per scheda)	***Pin 3,8***	***J17-J18***
	QUESTA FUNZIONE SU NGIO E' ATTIVA SOLO SU HARDWARE REV. 1.1		
NGPP	→ Ingressi FAST INPUT 1-4 (4 per scheda)	***Pin 1,2,3,4***	***J19***
NGMsX	→ Ingresso Tacca di Zero per ogni canale (2 per scheda)	***Pin 3,8***	***J22-J23***
NGQX	→ Ingresso Tacca di Zero per ogni canale (2 per scheda)	***Pin 3,8***	***J6-J8***

Proprietà principali

Card Index Indice della scheda nel bus
NGIO, NGPP, NGMsX – da 0 a 7
NGQx - 0
Channel Numero di canale di ingresso da associare
NGIO. NGMsX, NGQx – da 0 a 1
NGPP – da 0 a 3

Metodi

Name.get() as void
Update nelle registri di appoggio dei latch relativi al fronte di salita o discesa
(metodo da chiamare sempre prima di ogni lettura dei fronti UP e DN)

Name.clear() as void

Resetta i registri di appoggio dei latch relativi al fronte di salita o discesa
Questo metodo una volta chiamato, azzera gli stati dei registri UP e DN, ma se lo stato dell' ingresso si trova per es. ALTO, il relativo latch UP viene immediatamente settato. (viceversa per DN)

Variabili di sola lettura

Nome.stato	Contiene lo stato attuale (0 o 1) dell' ingresso selezionato (questa non è gestita ad INTERRUPT)
Nome.up	Contiene lo stato del LATCH del fronte di salita. Questa funzione è gestita in modo HARDWARE INTERRUPT dal sistema operativo
Nome.dn	Contiene lo stato del LATCH del fronte di discesa. Questa funzione è gestita in modo HARDWARE INTERRUPT dal sistema operativo

Esempio

Inserire un oggetto Fast Input di nome ***FastInpuit1***
Mettere il codice seguente nella main – Master Ciclo
Dichiarare le seguenti variabili:
RisingEdge1 char
FallingEdge1 char
State1 char

```
Fastinput1.get()              'get fastinput1
RisingEdge1=FastInput1.up     'controlla fronte di salita
FallingEdge1=FastInput1.dn    'controlla fronte di discesa
State1=FastInput1.inp         'legge lo stato
FastInput1.clear()            'resetta latch fronte salita e discesa
```

Eventi

nessuno

7 CLASSE MODBUS

La Macro classe MODBUS contiene la gestione SLAVE del protocollo MODBUS RTU e TCP/IP

Hardware ***Tutti (NG35 e Pec70 per TCP/IP)***

7.1 CModbus.vco – Salve Modbus RTU/TCPIP

Modbus RTU RS232

Proprietà principali

Nodo Nodo slave
BaudRate baud rate inserire valori concordi con periferiche esterne slave
PtData() Registri Unsigned Integer dello slave messi a disposizione del master. Indirizzo di partenza 0
Max Len Data Dimensione del buffer PtData
TimeOut Tempo in millisecondi interrogazione del master
Questo valore deve essere di norma inferiore al timeout impostato sul master

Metodi

Nessuno

Vengono gestite le seguenti richieste MODBUS RTU:

Function Code 3 Read Multiple Registers
Function Code 6 Preset Single Registers
Function Code 16 Preset Multiple Registers

Eventi

Nessuno

Modbus TCP/IP

Proprietà principali

Nodo Nodo slave
IpAddress Indirizzo IP dello slave es. “10.0.0.80”
Service Port Porta di servizio (default 502 dedicata la ModBus TCPIP)
PtData() Registri Unsigned Integer dello slave messi a disposizione del master. Indirizzo di partenza 0
Max Len Data Dimensione del buffer PtData

Metodi

Nessuno

Vengono gestite le seguenti richieste MODBUS TCP/P:

Function Code 3 Read Multiple Registers
Function Code 4 Read Input Registers
Function Code 6 Preset Single Registers
Function Code 16 Preset Multiple Registers

Eventi

Nessuno

8 CLASSE MOTOR CONTROL PLUS

La Macro classe MotorControlPlus è una famiglia di oggetti che rappresentano un upgrade di quelli presenti nella MotorControl descritta in precedenza. Sono oggetti più evoluti rispetto ai precedenti e sono da preferire nelle nuove applicazioni in quanto gli altri tenderanno a non essere più supportati.

Hardware ***Tutti***

8.1 CobjPos.vco – Posizionatore MONOASSE

Rappresenta l'evoluzione dell'oggetto MonoAx, a differenza del quale, invece di lavorare virtualmente su di una variabile di uscita, permette opzionalmente di selezionare direttamente il tipo di asse che vogliamo pilotare (stepper, DS402, analogico).
L'altra differenza importante è che questo è derivato dall'oggetto CobjInterpola e quindi presenterà tutte le sue caratteristiche principali (funzioni, buffer movimenti, rampe, ecc.).

Proprietà

NOTA: le proprietà in maiuscolo non sono modificabili a livello di run-time.

N. TRATTTI	Numero di tratti nel buffer movimenti
Vper	Percentuale velocità impostata (modificabile durante i movimenti)
Div. Vper	Divisore per Vper
AccQstop	Decelerazione per funzione qstop
Acc	Accelerazione / decelerazione
RzeroMode	Tipo di ricerca di zero. Vedi dettagli più avanti
RzeroOffset	Movimento asse dopo la ricerca di zero
RzeroPreset	Quota di preset dopo il posizionamento a offset
RzeroVel	Velocità ricerca di zero
RzeroVelf	Velocità ricerca di zero fine
RzeroAcc	Accelerazione/decelerazione ricerca di zero
Msof	Impulsi a giro
Dsof	Sviluppo di un giro, negativo per invertire la direzione positiva dell'asse
LimitN	Limite negativo extra-corsa software
LimitP	Limite positivo extra-corsa software
Gioco	Valore di recupero gioco
Vgioco	Velocità recupero gioco
MsofV	Moltiplicatore volantino
DsofV	Divisore volantino, negativo per invertire la direzione
qvola	Incremento volantino
pc	Quota posizionatore prima del volantino
qt	Quota posizionatore dopo somma con volantino
qi	Quota posizionatore in impulsi (compresi volantino e recupero gioco)
RZERO ENABLE	Inserisce il codice per la ricerca di zero (utilizzato per diminuire la memoria codice)
AXIS TYPE	Definisce il tipo di asse associato.

-1	nessun tipo di asse gestito
0	nessun tipo di asse gestito ma vengono chiamate le funzioni di interfacciamento obj.ax_xxx per la personalizzazione dell'asse (chiedere all'assistenza per eventuali dettagli)
1	CANOPEN (usare un oggetto di CstdCanopen)
2	STEP/DIR (NG-PP, NGM EVO, NGQ)
3	STEP/DIR SLAVE (slave CanOpen NGM EVO o NGQ)
4	ANALOGICO (usare un oggetto di PidPlus)

VTB AXIS OBJECT	Nome dell'oggetto che gestisce l'asse (solo con AXIS TYPE 1 e 4)
PDO NAME	Nome della variabile associata al pdo quota interpolata (solo con AXIS TYPE 1 e 3)
STEP CHANNEL	Canale dell'asse step sull'hardware (0, 1, 2….) (solo con AXIS TYPE 2 e 3)
STEP NODE	Nodo scheda NGM EVO o NGQ slave (solo con AXIS TYPE 3)

Metodi

function enable() as void
Abilita asse

function disable() as void
Disabilita asse

function preset(q as long) as void
Preset alla quota q. Per azzerare solo il volantino eseguire un preset alla quota attuale: obj.preset(obj.qt)

function move() as char
Test movimento in corso.

function stop() as void
Stop asse con attesa fine movimento

function fstop() as void
Stop asse senza attendere la fine movimento

function qstop() as void
Come fstop ma utilizza AccQstop

function StartHome() as void
Start ricerca di zero. Per sapere quando è terminata testare status_rzero.

function StopHome() as void
Interrompe la ricerca di zero.

function moveto(vel as long, ferma as char, q as long) as char
Movimento con fermata impostata

vel velocità
ferma 0 - Non ferma sul punto precedente
1 - Ferma sul punto precedente
q quota

Ritorno 0 = non c'e' posto nel buffer
1 = movimento inserito nel buffer

function lineto(vel as long, q as long) as char
Movimento con fermata calcolata automaticamente. In pratica l'asse si ferma tra due tratti solo se c'e' un'inversione di direzione, altrimenti prosegue senza fermarsi.

vel velocità
q quota

Ritorno 0 = non c'e' posto nel buffer
1 = movimento inserito nel buffer
-1 = movimento 0 sull'asse

Eventi

Nessuno

8.1.1 Bit interfacciamento I/O

Questi sono i bit predefiniti da associare agli ingressi fisici dell'hardware utilizzato.

Nome Bit	Descrizione
ext_fcn	Fine corsa negativo
ext_fcp	Fine corsa positivo
ext_fcz	Sensore ricerca di zero
ext_tacca	Tacca di zero

8.1.2 Status Bit

Nome Bit	Variabile	Descrizione
status_rzero		Ricerca di zero in corso
status_home		Viene settato quando l'asse ha eseguito la ricerca di zero
status_enable		Viene settato quando l'asse è abilitato
ErrorLimitN	Error.0	Limite negativo intervenuto
ErrorLimitP	Error.1	Limite positivo intervenuto

8.1.3 Ricerca di zero

La ricerca di zero viene eseguita in base al parametro RzeroMode. Una volta eseguita, l'asse viene spostato del parametro RzeroOfset e quindi eseguito il preset con la quota contenuta nel parametro RzeroPreset.
Ogni bit del parametro RzeroMode ha un proprio significato e si possone eseguire varie combinazioni. Le velocità sono impostate con RzeroVel (veloce) e RzeroVelf (lento) e viene sempre utilizzata l'accelerazione RzeroAcc.

RZERO_MODE		DIREZIONE INIZIALE	SEQUENZA
0	Ricerca su sensore attivo	Indietro (negativo)	- Indietro veloce fino al sensore di zero - Avanti lento per uscire dal sensore - Indietro lento fino al sensore - Posizionamento alla quota di offset
1		Avanti (positivo)	- Avanti veloce fino al sensore di zero - Indietro lento per uscire dal sensore - Avanti lento fino al sensore - Posizionamento alla quota di offset
2	Ricerca su sensore attivo e tacca di zero	Indietro (negativo)	- Indietro veloce fino al sensore di zero - Avanti lento per uscire dal sensore - Indietro lento fino al sensore - Proseguimento fino alla prima tacca - Posizionamento alla quota di offset
3		Avanti (positivo)	- Avanti veloce fino al sensore di zero - Indietro lento per uscire dal sensore - Avanti lento fino al sensore - Proseguimento fino alla prima tacca - Posizionamento alla quota di offset
4		Indietro (negativo)	- Indietro veloce fino al sensore di zero

	Ricerca su sensore disattivo		- Avanti lento per uscire dal sensore - Posizionamento alla quota di offset
5		Avanti (positivo)	- Avanti veloce fino al sensore di zero - Indietro lento per uscire dal sensore - Posizionamento alla quota di offset
6	Ricerca su sensore disattivo e tacca di zero	Indietro (negativo)	- Indietro veloce fino al sensore di zero - Avanti lento per uscire dal sensore - Proseguimento fino alla prima tacca - Posizionamento alla quota di offset
7		Avanti (positivo)	- Avanti veloce fino al sensore di zero - Indietro lento per uscire dal sensore - Proseguimento fino alla prima tacca - Posizionamento alla quota di offset
8	Ricerca solo su tacca di zero	Indietro (negativo)	- Indietro lento fino alla prima tacca - Posizionamento alla quota di offset
9		Avanti (positivo)	- Avanti lento fino alla prima tacca - Posizionamento alla quota di offset
18	Ricerca su sensore attivo e preset a qtacca *	Indietro (negativo)	- Indietro veloce fino al sensore di zero - Avanti lento per uscire dal sensore - Indietro lento fino al sensore - Preset a quota tacca - Posizionamento alla quota di offset
19		Avanti (positivo)	- Avanti veloce fino al sensore di zero - Indietro lento per uscire dal sensore - Avanti lento fino al sensore - Preset a quota tacca - Posizionamento alla quota di offset
22	Ricerca su sensore disattivo e preset a qtacca *	Indietro (negativo)	- Indietro veloce fino al sensore di zero - Avanti lento per uscire dal sensore - Preset a quota tacca - Posizionamento alla quota di offset
23		Avanti (positivo)	- Avanti veloce fino al sensore di zero - Indietro lento per uscire dal sensore - Preset a quota tacca Posizionamento alla quota di offset
32	Ricerca disabilitata	Zero su enable	Tutte le volte che si invia un comando di abilitazione l'asse viene presettato alla quota RZERO_PRESET
64		Encoder assoluto	Al comando di abilitazione il sistema si presetta sulla quota dell'asse
128	Ricerca esterna (da codice VTB)	La ricerca di zero è affidata a codice esterno in vtb. Viene messa ad 11 la variabile RzeroStato. Il codice esterno dovrà provvedere ad aggiornare i bit della status (status_home, status_rzero) e azzerare RzeroStato.	

* La lettura della quota tacca è affidata al codice VTB. Viene messa a 51 la variabile RzeroStato. Il codice esterno dovrà scrivere in qtacca il valore letto dal driver (in impulsi) e impostare a 55 la stessa RzeroStato per far terminare la procedura.

8.1.4 Selezione tipo di asse

AXIS TYPE	
-1	L'oggetto non gestisce nessun tipo di asse ma lavora virtualmente sulla quota di uscita (proprietà qt)
0	L'oggetto non gestisce nessun tipo di asse ma richiama le funzioni di personalizzazione. Impostando a 0 AXIS TYPE il compilatore genererà degli errori di funzione non trovata. Contattare la PROMAX per la documentazione relativa alla personalizzazione dell'asse.
1 - CANOPEN DS402	- Inserire un oggetto della classe CstdCanopen che gestisce l'asse associato - Impostare VTB AXIS OBJECT con il nome dell'oggetto inserito - Eseguire la configurazione CanOpen - Impostare PDO NAME col nome della variabile associata al PDO della quota interpolata - STEP CHANNEL e STEP NODE non utilizzati
2 – STEP/DIR	- Non occorre inserire ulteriori oggetti - Impostare STEP CHANNEL con l'indice del canale sulla scheda NG-PP, NGM EVO, NGQ associato all'asse - VTB AXIS OBJECT, PDO NAME e STEP NODE non utilizzati
3 – STEP/DIR su slave Canopen	- Non occorre inserire ulteriori oggetti - Eseguire la configurazione CanOpen - Impostare PDO NAME col nome della variabile associata al PDO della quota interpolata - Impostare STEP CHANNEL con l'indice del canale sulla scheda slave Canopen NGM EVO o NGQ associato all'asse - Impostare STEP NODE col numero nodo dello slave CanOpen - VTB AXIS OBJECT non utilizzat0
4 – ANALOGICO	- Inserire un oggetto della classe CPidPlus che gestisce l'asse associato - Impostare VTB AXIS OBJECT con il nome dell'oggetto inserito - Impostare eventualmente i parametri del PID - PDO NAME, STEP CHANNEL e STEP NODE non utilizzati

8.2 CPidPlus.vco – Regolazione PID

La Macro classe PidPlus consente la regolazione PID di un asse analogico. E' una evoluzione dell'oggetto FiltroDigitale che, a differenza del precedente, gestisce direttamente l'hardware utilizzato

Proprietà

NOTA: le proprietà in maiuscolo non sono modificabili a livello di run-time.

EnablePid Abilita il controllo pid senza attivare il relè di abilitazione
Kp, Ki, Kv, Kd Parametri PID
Err_Sat Parametro di saturazione sommatoria errori per Ki
NG ENC CHANNEL Indice canale encoder sulla scheda utilizzata
NG DAC CHANNEL Indice canale uscita analogica sulla scheda utilizzata
ENABLE KP, KI, KV, KD Abilitazioni per ottimizzare la memoria codice
Divisore **Divisore per tutti i parametri PID**
Dir Polarità uscita analogica (0 o 1)
ServoErr Parametro di soglia servo error
TServoErr Tempo per intervento servo error (in mSec)
EnableDelay Ritardo tra attivazione relè e attivazione PID (in mSec)
Err Stato servoerror ad 1 errore raggiunto (solo in run time)
Post Demand Position (solo in run time)
Posr Actual Position (solo in run time)

Metodi

function enable() as void
Abilita asse. Attiva il relè di abilitazione e il controllo PID.

function disable() as void
Disabilita asse. Disattiva il controllo PID e il relè di abilitazione

Eventi
Nessuno

8.3 CRotaryKnife.vco – Taglio rotativo

Gestisce il taglio rotativo con rampe trapezioidali o sinusoidali e gestione extra velocità sinusoidale.
Rappresenta un upgrade dell'oggetto CtaglioRot.
Una volta impostati i parametri occorre richiamare la funzione di calcolo della camma (TrapUpdate o SinUpdate) di lavoro. La nuova camma così calcolata verrà attivata all'inizio del giro successivo.
Come il precedente oggetto lavora come slave rispetto ad una quota master proveniente dall'esterno. Per questo motivo eventuali arresti in fase dovranno essere implementati da codice VTB.

Proprietà

Enable	Abilita il funzionamento dell'oggetto
Len	Lunghezza taglio in mm (0.1mm, 0.01mm, ecc)
Imaster	Impulsi encoder master
Emaster	Sviluppo encoder master in mm (0.1mm, 0.01mm, ecc)
Islave	Impulsi encoder slave (rotare knife)
Diam	Diametro slave (rotare knife) in mm (0.1mm, 0.01mm, ecc)
Async	Angolo di sincronismo in gradi
Ksync	Velocità slave di sincronismo in % (100=sincronismo)
Vextra	Extra velocità di sincronismo in % (0=disabilitata)
StartExtra	Punto in % all'interno dell'angolo di sincronismo dove iniziare l'extra velocità
StopExtra	Punto in % all'interno dell'angolo di sincronismo dove finire l'extra velocità
Shift	Quantità di impulsi master di sfasamento. Viene decrementato ad ogni campionamento del valore Vshift fino ad azzerarsi
Vshift	Velocità di sfasamento in impulsi master a campionamenti
Master	Variabile contenente il valore del conteggio dell'encoder master
Slave	Variabile dove viene scritto il valore degli impulsi slave (da passare all'asse)
Cam Len	Numero di punti della camma di lavoro
state	Stato dello slave (sola lettura): **0** non inizializzato 1 fase di raggiungimento velocità di sincronsimo **2** fase di sincronismo **3** fase di ritorno alla velocità di riposizionamento o attesa
qmaster	Posizione corrente dello slave in millesimi di grado (sola lettura)
vslave	Velocità corrente dello slave in impulsi a campionamento (sola lettura)

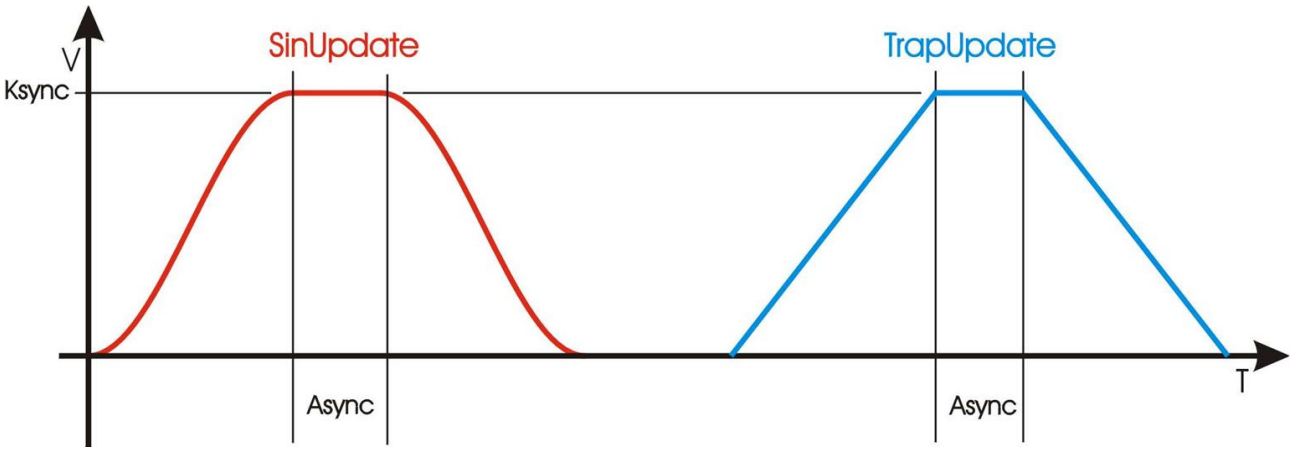
ATTENZIONE!!!
Le proprietà che rappresentano misure lineari devono avere TUTTE la stessa unità di misura (centesimi di millimetro, millesimi di millimetro, ecc,).

Metodi

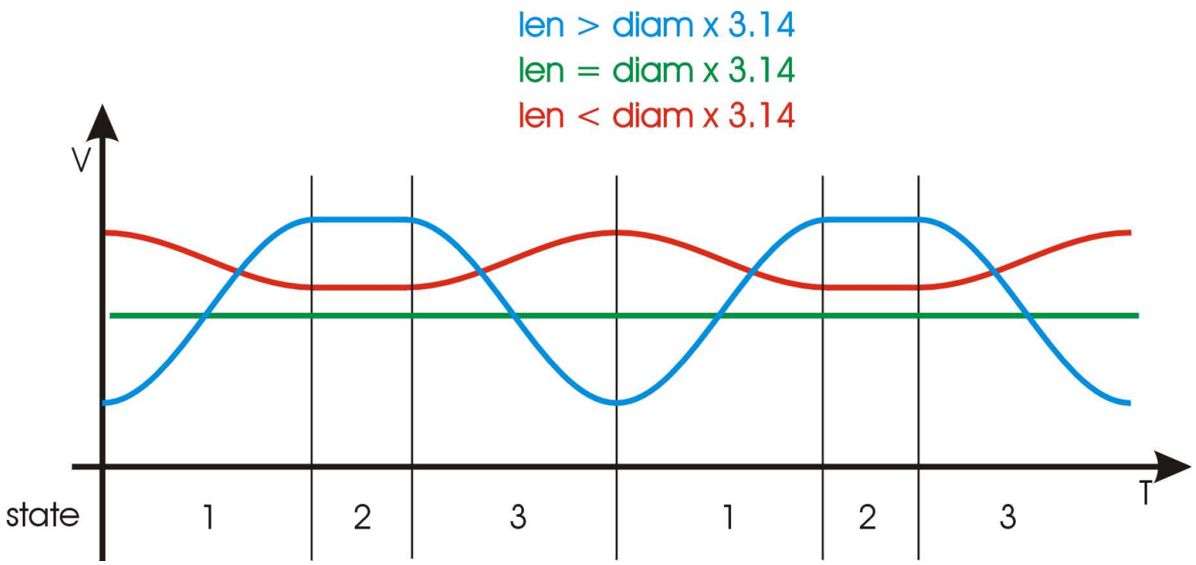
function reset() as void
Ripristina i valori iniziali dei contatori e disabilita l'oggetto (enable=0). L'ultima camma calcolata rimane comunque in memoria. Per ripartire occorre rimettere ad 1 la proprietà enable.

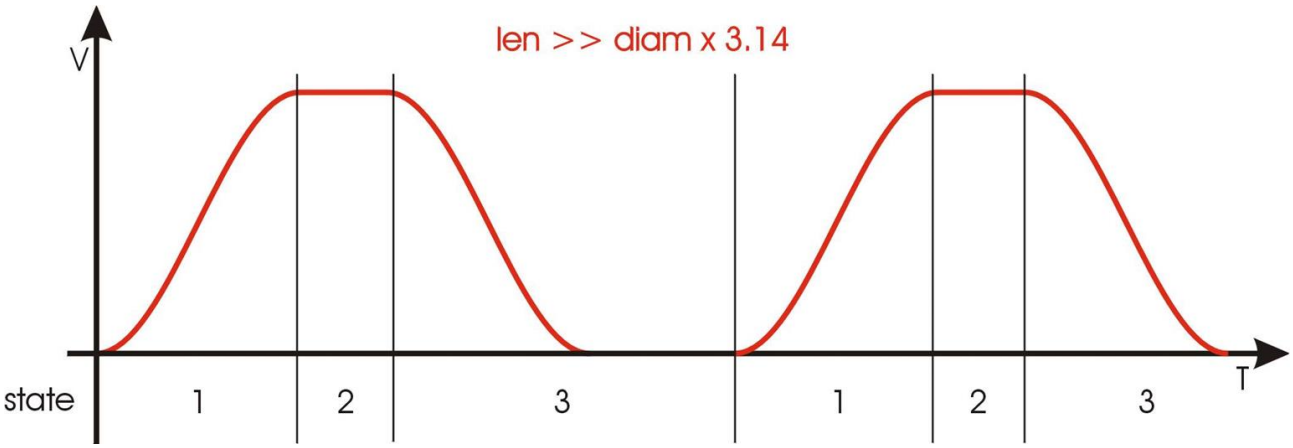
function TrapUpdate() as void
Aggiorna i parametri calcolando la camma con forma trapezioidale.

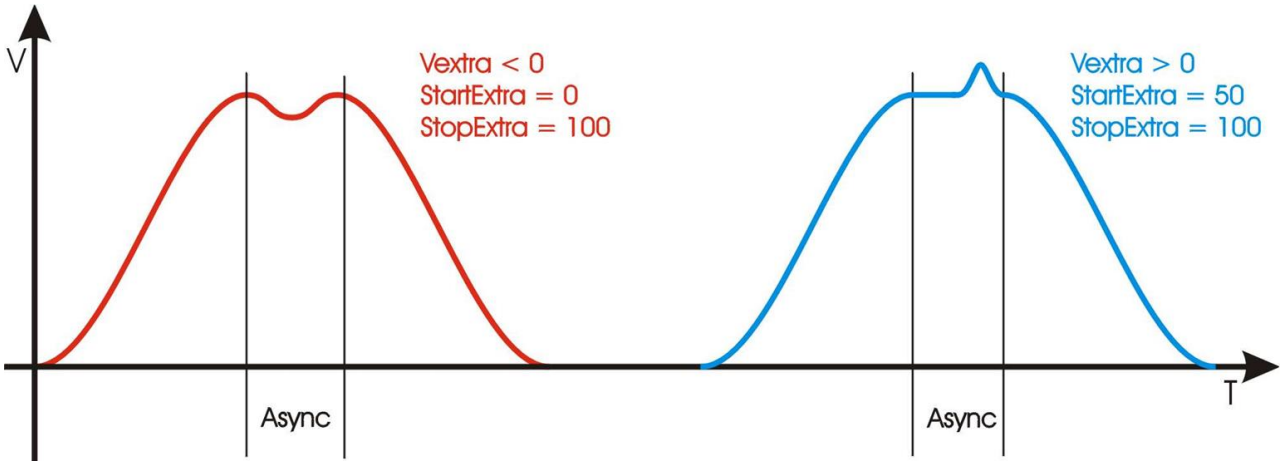
function SinUpdate() as void
Aggiorna i parametri calcolando la camma con forma sinusoidale.



8.3.1 Esempi







8.3.2 Rifasamento tramite tacca

Per ottenere il rifasamento del asse rotativo con una tacca di riferimento occorre aggiungere del codice esterno in base alle esigenze di funzionamento. L'oggetto mette a disposizione due proprietà: Shift e Vshift.
Il primo imposta gli impulsi master di correzione che vogliamo introdurre, il secondo dice quanti impulsi correggere ad ogni campionamento (TaskPlc). Consente cioè di “spalmare gli impulsi” nel tempo per evitare una brusca correzione. Una volta impostato un valore di Shift, ad ogni campionamento viene effettuato uno sfasamento pari a Vshift fino ad azzerare il valore di Shift. Il tempo di intervento dello sfasamento impostato sarà quindi pari a Shift/Vshift campionamenti.

Sommario