ATEE - 2004

MULTIPLE AXIS DISTRIBUTED MOTION CONTROL SOLUTIONS WITH INTELLIGENT DRIVES, USING CANOPEN COMMUNICATION PROTOCOL

Iulian Iacob, Augustin Ignat, Aurelian Sarca, Liviu Kreindler

Abstract: This paper describes the concept of multi-axis distributed motion control using intelligent servo drives connected over a CAN network. The CANopen implementation is used as the higher layer communication protocol. The trajectory generation is not any more the task of a single host, as it is shared between the CAN host and the CAN servo nodes, which also include the position / speed / current control loops. We will show the advantages of the CANopen protocol

INTRODUCTION

As far as multiple axis control is concerned, the standard and classical approach was to consider dedicated multi-axis motion controllers, either PC-based or standalone. This single controller included both the trajectory generation and the higher position control loop. The velocity or current command was sent to simple servo drives, either analog or digital, via a ±10 V analog signal. The feedback signals (usually position read from an encoder / resolver / potentiometer) have to come from the motor to the controller and probably have to travel a long way to get there. Also, as the servo drives have to be relatively close to the motor, the command signals from the controller might be pretty long. The noise levels can become quite great and can negatively influence not only the accuracy of the system, but also its reliability. Such low voltage signals tend to be very sensitive to noise in harsh operating environments. Also the cost of the wiring can play a significant part in the overall system cost.

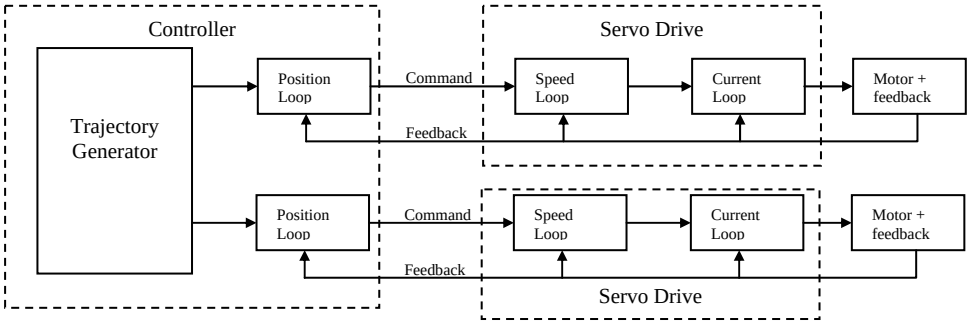

Fig. **1.** Classical centralized control scheme

Another thing to keep in mind in this system structure is the fact that changing or adding a new axis can prove very difficult due to a fixed hardware platform. Another very important feature in a top control system is the possibility to perform self-diagnostics and to signalize potential problems even before they actually happen. This centralized system does not allow for a high extent of diagnostic and / or performance analysis.

DISTRIBUTED CONTROL

The concept of distributed control is based both on the development of the network components and solutions and on the development of ever more powerful digital control solutions. Servo drives equipped with digital signal processors (DSP) allow the execution of complex control algorithms and trajectory generation / interpolation locally at the drive level.

This way the host controller will only be responsible to issue high level commands to the intelligent servo drives. The structure of the system is presented in figure 2.

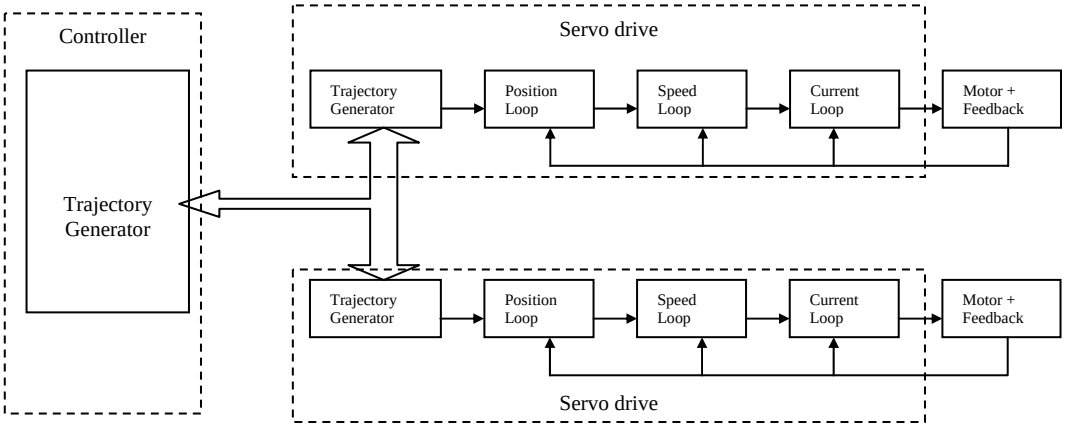

Fig. 2. Distributed control scheme

The commands from the CAN master may range from sending position / speed set points that describe a final trajectory to calling different functions already stored inside the memory of the intelligent servo drive (i.e. homing functions). The local intelligence of the drives allows them to react (in a preprogrammed manner) to various error or fault conditions. This way the host controller will be promptly informed of the error.

INTELLIGENT DRIVES WITH INTEGRATED CAN BUS INTERFACE

The master and the slave drives are interconnected on a digital communication network. One of the possibilities is having as a physical layer a medium speed serial connection like CAN (Controller Area Network) over which they communicate using the CANopen protocol. In the CANopen implementation, CAN is a differentially driven two-wire bus line with common return [1]. Maximum transmission speed is 1 Mbit/s. The hardware specifications of the CAN bus require a message to be validated by every active node on the bus. Furthermore, data integrity checking is performed at the hardware level in the CAN controller, insuring a message error rate of less than $4.7 * 10^{-11}$.

A CAN messages consists in an identifier (called COB-ID – communication object identifier) and up to 8 bytes of data. Being a system that does not require a communication master (a device that is responsible to poll the other devices in order to communicate), a collision avoidance system is implemented based on the value of the COB-ID – in case two different devices are simultaneously trying to send a message, the device that sends the message with the smaller COB-ID wins and the other device automatically transits in receive mode and waits for the bus to be idle again before attempting another transmission.

The CANopen protocol provides means for a master controller to configure and set-up a slave device using standardized messages. There are two distinct layers, a communication layer that provides the kernel for the communication objects and network management [2], and a device profile that defines the behaviour of motion control devices [3].

The two layers were implemented on a Technosoft IDM640-8EI intelligent servo drive [4] built around a Texas Instruments TMS320LF2407A DSP [5]. The CANopen communication profile was added to the pre-existing TML language of the IDM640-8EI intelligent servo drive.

CANOPEN IMPLEMENTATION

The communication profile consists in a collection of objects called the object dictionary that can be accessed via specific CANopen messages. The network management protocol (NMT) is responsible with the configuration of the drive from the network point of view, as the statuses that can be assigned to the drive define what other protocols are active at a certain moment. Through this network management protocol the drive can be commanded to enter the operational, pre-operational and communication reset states. Another purpose of the NMT protocol is to test / check the presence of a certain drive in the network, acting this way as a powerful tool of fault detection. There are 2 ways to implement this test, in either active, or passive way, from the CAN host point of view: Node Guarding (when the master actively polls all the slaves at certain time intervals) or Heartbeat (when all the drives are responsible to send, also at regular intervals, their status to the master).

A powerful feature of the CANopen and its implementation is the existence of a synchronization mechanism provided by the SYNC protocol. One of the drives in the network assumes the role of synchronization master, which means it is the one responsible to send the SYNC message. This is a high priority COB-ID that carries no data and is sent at regular time intervals. The usual interval of SYNC message sending is 20 ms. With the help of the TIMESTAMP message we can achieve a perfect synchronization of all the drives in the network. This synchronization is crucial in performing coordinated multi-axis movements. Without this mechanism the control loops of the drives in the network would drift one from the other in time, reaching a point were a coordinated multi-axis movement would not be possible due to the different natural frequency of the quartz crystals that provide the base frequency of the drives. The TIMESTAMP message is sent by the synchronization master after it has provided the SYNC message on the network. All the drives will receive simultaneously the TIMESTAMP message that contains the absolute time of the synchronization master and will perform corrections to their own internal absolute time in order to ensure perfect synchronization.

Every drive in the network will be able to signalize an internal or external error condition detected through the EMCY channel. This is a dedicated message object that will be sent with standardized and / or specific error codes.

The master is able to configure and parameterize every drive in the network before actually commanding them to perform any active operation through the SDO protocol (Service Data Object). This is a confirmed protocol initiated by an SDO client (in our case the master) and works bidirectional – the client may perform a download to the SDO server (the intelligent drives) or request data through an upload from the SDO server. The SDO mechanism allows the transfer of large amount of data, from simple data types to complex data types such as programs to be executed after the download. Since SDO is a confirmed protocol, it incorporates error detection and transfer abortion mechanisms that insure the data sent by either the server or the client will arrive without errors. Abortion codes are provided in case the object dictionary entries accessed are not available on the addressed client or the object dictionary entry contains a data incompatible with the requested access type.

Another major data exchange mechanism is the PDO protocol (Process Data Object). These are non-confirmed messages targeted to the real time data exchange. As each CAN message carries only 8 bytes of data, the PDO’s have to be pre-programmed to specific tasks or specific system variables. They can be either sent (transmit PDO - TxPDO) or received

(receive PDO - RxPDO) by the drive. The receive PDO's usually carry command information such as position / speed set points, target position to be achieved through a certain type of interpolation, other commands grouped in a so-called command word, commands to activate / change IO status, etc. The transmit PDO's can carry information regarding the drive current position, speed and / or status and digital inputs status. The available position reference generation methods include profile position, interpolated position (PVT – position, velocity, time mode – cubical interpolation, where the master controller can achieve a multi-axis coordinated motion by de-composing the whole movement on each axis and segmenting the movement by sending points of the real movement [6] or PT – position and time mode – linear interpolation) or point to point moves.

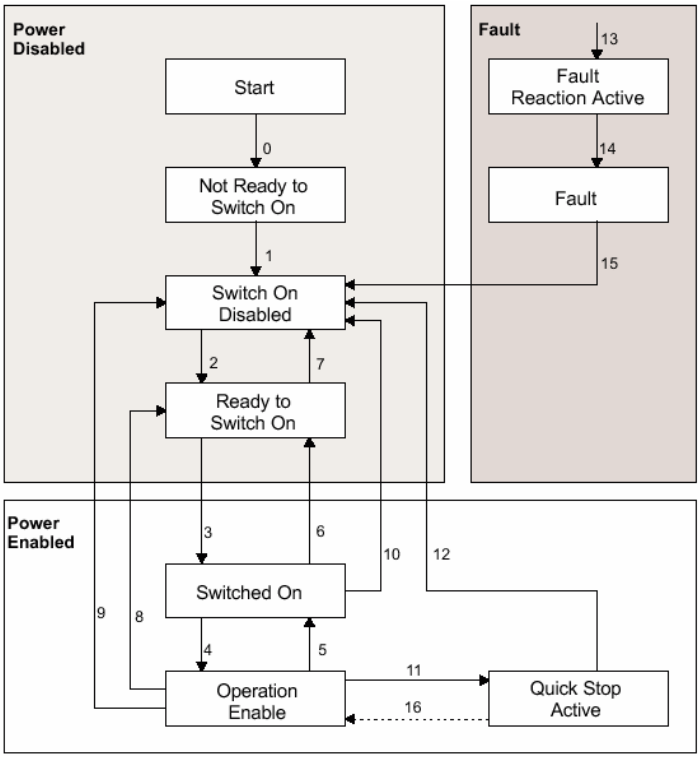

Fig. 3. CANopen states diagram

Using the control word the master can issue commands to the slave in order to switch it on or off (from the motor / motion control point of view), as seen in figure 3. Different statuses are defined with different parts of the drive being energized. Each transition from the diagram in figure 3 represents either a control word change (in this case the transition is requested by the master) or is caused by an internal or external factor (entering in a Fault state of in a Quick Stop Active state). Each such a transition is confirmed / signaled to the master by sending a PDO with the new status.

At a PDO reception, the drive, apart from setting the corresponding parameters or variables, can also trigger the call of some custom routines (such as homing or any other user defined functions).

As a conclusion, the distributed approach to motion control provides both precise control and diagnostic capabilities, saves valuable resources by getting rid of the analogue wiring and insures unrivalled flexibility for the systems integrators.

REFERENCES

[1] CAN Physical Layer for Industrial Applications, CAN in Automation, 1994
[2] CANopen Application Layer and Communication Profile, CiA Draft Standard 301, CAN in Automation, 2000
[3] CANopen Device Profile for Drives and Motion Control, CiA Draft Standard Proposal DSP402, CAN in Automation, 1998
[4] IDM640 Technical Reference, IDM640-8EI User Manual, Technosoft, 2004
[5] TMS320LF/LC240xA DSP Controllers Reference Guide, System and Peripherals, Texas Instruments, Literature Number SPRU357B, 2001
[6] Bosteels, Jan, "Coordinated Multi-Axis Motion Control via CAN bus", 8th International CAN Conferrence, 2002