NUMERIC CONTROL OF CENTRE LATHE

Cătălin – Iulian CHIVU
Transilvania University of Braşov, Romania

Abstract. Traditional CNC machines are expensive, complicated, with high precision, and typically only found in large manufacturing companies that can afford them. To implement these machine-tools it is necessary to train the employees. Not always a university can afford to let the students practice on such machines, to direct work on them. Thus, at the beginning of the learning process in this field, it is important that the students understand mechanical and driving systems, or how can be implemented a Numerical Control on such a complex machine. Present paper presents the step that should be followed to transform a classical lathe into a numerical controlled one.

Keywords: lathe, numerical control, programming, Delphi

1. Introduction

Computer numerical control is a very broad term that encompasses a variety of types of machines all with different sizes, shapes, and functions. But the easiest way to think about CNC is to simply understand that it's all about using a computer as a means to control a machine [1].

In NC machines, the tool/table is moved automatically, according to the program fed into the control computer. Instead of approximately set stoppers, numerals and letters are used to move the tool/table precisely, through computerized actuators. Selection of speed and feed, switching on and off, and tool/table motions, are automated by a program, which depends upon the workpiece material, shape, and operation. [2].

All the above aspects increase the cost of such machine-tools and make the teaching process very difficult because of very expensive maintenance. Thus, to have an appropriate teaching material it was used an existing lathe (a smaller variant of a 400 centre lathe). Thus, students are able to identify the main mechanical characteristics but they also are able to see the connection between mechanical parts, electrical drive and control system.

Present paper is focused on implementing the control system on an existing such a system.

2. Construction of existing lathe

2.1. Mechanical components

The mechanical structure of the lathe is similar to any centre lathe: spindle head (headstock), feed gear box, lead screw (there is no feed shaft), longitudinal and transversal saddles and tool post (positioned on a transversal saddle).

The spindle head is driven by a three-phase, asynchronous AC motor, of 1 kW through belt gearing transmission. Belt tension can be achieved by a lever that changes the distance between motor shaft and headstock shaft, keeping them parallel, by changing the angle of inclination of the engine in a plane normal to the engine's axis. Transmission ratio can be determined by choosing an appropriate combination of pairs of pulleys, between the three mounted on headstock shaft and two mounted on motor shaft.

Longitudinal feed is made using a lead screw. The screw is driven either mechanical, using reducing gear that connects motor to headstock shaft, or manually, using a lever with a scale that allows an accurate positioning. There is a coupling device, manually operated, that allows to switch between manual and mechanical feed. This coupling device is positioned on the left of the lathe and connects lead screw to reducing gear device. The lead screw determines the longitudinal feed step (4 mm/rotation). Transversal feed and tool post feed are done also using screw mechanisms. Step for transversal is 5 mm/rotation, movement is manually operated through a scaled lever.

To be able to transform a centre lathe structure into a numerical controlled one there were done some changes:
- elimination of reducing gear connected to longitudinal feed mechanism;
- connect a step-by-step motor, through mobile coupling device, to lead screw of longitudinal feed;
- connect a step-by-step motor through mobile coupling device, to transversal feed screw.

In addition to changes made so far, is already emerging the need for additional changes as those presented in the conclusions of the present paper.

2.2. Electrical drives

Spindle head is driven by a three-phase, asynchronous AC motor, of 1 kW. The rated speed of such a motor depends on load, friction loss in kinematic chain and bearings. Thus, the headstock speed is manufacturing parameter, which depends on the task. Because of this it is mandatory an auxiliary device to synchronise the headstock speed with longitudinal and transversal movements [3].

To obtain numerical control it was necessary to implement an electric drive on both longitudinal and transversal shafts [4].

On longitudinal shaft was connected a step-by-step motor with 1.8 degree step and a torque of 8 Nm, thus a rotation being equivalent with 200 steps. Besides, the step of lead screw is 4 mm, thus a motor step is equivalent to 0.02 mm (20 µm) longitudinal displacement.

Transversal shaft is driven by a step-by-step motor with 1.8 degree step and a torque of 2 Nm, thus a rotation being equivalent with 200 steps. Besides, the step of transversal screw is 5 mm, thus a motor step is equivalent to 0.025 mm (25 µm) longitudinal displacement.

2.3. Electronic circuit

The headstock motor is actuated from the building three-phase circuit, using three fusible fuses and a three-phase switch (stop, forward rotation, backward rotation).

For longitudinal and transversal shafts on witch was implemented numeric control, there were developed and implemented the command electronic circuits for the two step-by-step motors. Actuating circuit for longitudinal motor has an integrated module that includes AC alimentation. Actuating circuit for transversal motor has two modules. First includes: TTL (Transistor-Transistor Logic) interface, electronics galvanic isolated and amplification of the motor winding current command. Second module includes: DC adjustable stabilized power supply, set to 24 V, 5 A. The interface of first module requires a distinct DC power supply source, set at 5 V.

Both actuating circuits of the two step-by-step motors have the same standard interface, even if the conceptual part is totally distinctive. Both circuits have two inputs: step input and sense input. Step input command the motor step when voltage is switched from 0 V to 5 V. When the voltage is reversely switched (from 5 V to 0 V) there is no action. Sense input determines the rotation sense of the step-by-step motor for next steps. If voltage applied to this input is 0 V the motor is rotating to the left and when the voltage is 5 V the rotation is to the right. These two circuits help the user to command both motors with one step either to the left or right. Voltage levels are compatible both with TTL and CMOS (Complementary Metal Oxide Semiconductor) technology. This determines the connection between logic command inputs to parallel port of the computer.

The command circuit is shown in figure 1.

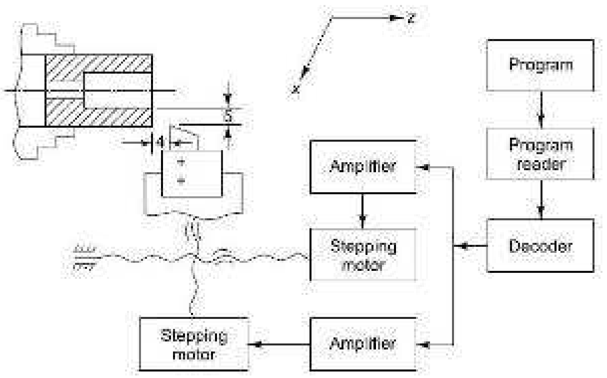

Figure 1. Command system of NC lathe

2.4. Computer interface

Command circuits of the two step-by-step motors are connected to a computer, through parallel port (longitudinal, transversal steps and sense). First step of this connection is to settle the reference electric potential. This is done by connecting the computer ground to ground of the two command circuits.

Because one of the command circuits requires a 5 V supply voltage, this circuit will be connected directly to supply source of the computer.

The four inputs of the command circuits will be connected to four outputs of the parallel port, taking over the computer controlled voltage levels.

Parallel port (Table 1) has three component ports: data, state and control port. Standard connector is DB25. Data port has 8 data pins, preset as outputs, corresponding to b0, b1, ... , b7 data bits. State port has 5 data pins, preset as inputs. Control port has 4 pins, configurable either as input or output.

To command the two circuits will be used four data bits, less significant: 0 bit (b0 = pin 2) for longitudinal step; 1 bit (b1 = pin 3) - longitudinal sense; 2 bit (b2 = pin 4) – transversal step; 3 bit (b3 = pin 5) – transversal sense. Software changes of these four bits determine the longitudinal and transversal displacements.

Table 1. Parallel port configuration

Pin No (DB25)	Signal name	Direction	Register - bit	Inverted
1	Strobe	In/Out	Control-0	Yes
2	Data0	Out	Data-0	No
3	Data1	Out	Data-1	No
4	Data2	Out	Data-2	No
5	Data3	Out	Data-3	No
6	Data4	Out	Data-4	No
7	Data5	Out	Data-5	No
8	Data6	Out	Data-6	No
9	Data7	Out	Data-7	No
10	Ack	In	Status-6	No
11	Busy	In	Status-7	Yes
12	Paper-Out	In	Status-5	No
13	Select	In	Status-4	No
14	Linefeed	In/Out	Control-1	Yes
15	Error	In	Status-3	No
16	Reset	In/Out	Control-2	No
17	Select-Printer	In/Out	Control-3	Yes
18-25	Ground	-	-	-

3. Computer command

The previous paragraph presented that command of the two movements imply the computer parallel port programming. To be able to obtain this, the author of present paper developed software application in Delphi programming environment [5]. The software is modular constructed. First step of the application is focused on programming the parallel port. This is done using three memory registers, on 8 bits, data register for data port, status register for status port and control register for control port.

Each of these three registers has an address according to a memory location (register) used either to read or write data. Base address of parallel port is 378 hex. Following two addresses correspond to status and, respectively, control register. Thus, there can be declared three constants, corresponding to these three addresses:

```
const
  Data = $378;
  Status = Data + 1;
  Control = Data + 2;
```

It should be mentioned that some of the parallel port pins are input pins (receive electrical signals from outside the computer and read them as logical values), some output pins (computer inside logical values are converted into signal output) and some have double function: input and output. Data port pins are all (8 pins) output pins.

Procedure for sending a positive integer number on 8 bits to specified memory address:

```
procedure TForm1.WritePort(PortValue,
DataValue:word);
begin
DataValue:=(DataValue*256)+DataValue;
  asm
  Mov ax,DataValue
  Mov dx,PortValue
  Out dx,ax
  end;
end;
```

For our application the port command should be done independently on each bit. This can be done either masking method or recalculation of value that is send to port (MyByte), based on the eight component bits (b0, b1, ..., b7). For a better understanding was chosen the second method:

```
procedure TForm1.CalculByte;
begin
MyByte:=0;
if (b0) then MyByte:= MyByte + 1;
if (b1) then MyByte:= MyByte + 2;
if (b2) then MyByte:= MyByte + 4;
if (b3) then MyByte:= MyByte + 8;
if (b4) then MyByte:= MyByte + 16;
if (b5) then MyByte:= MyByte + 32;
if (b6) then MyByte:= MyByte + 64;
if (b7) then MyByte:= MyByte + 128;
end;
```

Because output signals of lathe command should have a determined evolution (correlation between output signals and program commands) it is needed a synchronous actualisation of output signals (outputs should be changed only once before sending to port, otherwise some changes will not be actualised). This was done using the Timer. This element generates an event at each end of specific time period. Specific time period is an integer number parameter given in milliseconds. Timer generates an event used to compute the four bits of lathe command and value for parallel port and to send this value.

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
PasAvansLongitudinal;
PasAvansTransversal;
CalculByte;
WritePort(Data,MyByte);
i:=i+1;
if checkbox3.checked then EvaluareEroare;
end;
```

Sense of rotation for the two motors may be changed anytime. Application developed by the author has four procedures for changing the sense of rotation. Thus, the four procedures are:

- right rotation of longitudinal:

```
procedure TForm1.AvansLongitudinalDreapta;
begin
if not(b1) then
  begin
  PozInstLong:=PozInstLong-JocL;
   b1:=true;
  end;
end;
```

- left rotation of longitudinal displacement:

```
procedure TForm1.AvansLongitudinalStanga;
begin
if b1 then
  begin
  PozInstLong:=PozInstLong+JocL;
  b1:=false;
  end;
end;
```

- right rotation of transversal displacement:

```
procedure TForm1.AvansTransversalDreapta;
begin
if not(b3) then
  begin
  PozInstTransv:=PozInstTransv-JocT;
  b3:=true;
  end;
end;
```

- right rotation of transversal displacement:

```
procedure TForm1.AvansTransversalStanga;
begin
if b3 then
  begin
  PozInstTransv:=PozInstTransv+JocT;
  b3:=false;
  end;
end;
```

Because of the specific clearances of the coupling mechanisms, these parameters should be introduced in the program. Thus, it was noted with *JocL* the clearance for longitudinal coupling and, respectively, *JocT* the clearance for the transversal one. Values of these two parameters were included in the accordingly motor. These values were experimental determined.

The clearance introduces an error in the algorithm of computing the exact relative position of the motor. This is justified by the fact that, when the rotation reverses, for a few steps, proportional to the clearance, the motors rotates with no load. To compensate this error it is sufficient to add or subtract the value of this clearance, to the instantaneous position value.

To determine the logical value of two of four outputs it should be considered the time variable. Logical levels of step command output have to satisfy some minimum condition that regard time schedule. Step command signal is a pulse one. Signal starts at t0 moment, in to logic 0, and also at t0 switches between 0 logic to logic 1. Signal remains in logic 1 during a settled minimum period, until t1. At moment t1, signal switches between logic 1 back to logic 0, remaining there for a settled minimum period, until t2. Time between t1 and t0 characterises logic 1 (pulse width of 1 amplitude), time between t2 and t1 characterises logic 0 (pulse width of 0 amplitude) and time between t2 and t0 is the T period of signal (is a pulse signal). Moment t0 of one period is identical to moment t2 of the next period. To be able to detect the software changes at port output the 1 width of signal (time between t1 and t0) should be equal or greater to period of timer. Thus, result that minimum T period of step signal can't be less than twice timer period. The procedure that determines if the step signal for transversal displacement is allowed to be changed at the next update of port data is:

```
procedure TForm1.PasAvansLongitudinal;
begin
if dunuL>0 then
  begin
  b0:=true;
  dunuL:=dunuL - 1;
  perioadaL:=perioadaL - 1;
  end
else
  if perioadaL>0 then
    begin
    b0:=false;
    perioadaL:=perioadaL-1;
    end;
end;
```

The procedure that determines if the step signal for longitudinal displacement is allowed to be changed at the next update of port data is:

```
procedure TForm1.PasAvansTransversal;
begin
if dunuT>0 then
  begin
  b2:=true;
  dunuT:=dunuT - 1;
  perioadaT:=perioadaT - 1;
  end
else
  if perioadaT>0 then
    begin
    b2:=false;
    perioadaT:=perioadaT-1;
    end;
end;
```

In the above procedure variables are:

- *dunuL* – pulse width of 1 amplitude for longitudinal displacement;
- *perioadaL* – period of longitudinal step signal;
- *dunuT* – pulse width of 1 amplitude for transversal displacement;
- *perioadaT* – period of transversal step signal;

The relative unit of all four variables is timer characteristic period.

4. Software controller

Any software user has great expectations regarding the level of friendship of an informatics application interface. Thus, using software, nobody wants to do additional computing or unit transformation of the inputs or great effort to manage the interface. This is the reason why the software developed by the author of the present paper has, as inputs, all data given in *mm*. Solution of this problem is to simply multiply the desired position, computed in *mm*, with a constant that specify the number of steps per *mm* executed by the specific step-by-step motor:

```
const
  KT = 40;
  KL = 50;
```

where *KT* is number of steps/mm along transversal direction and, respectively, *KL* – steps/mm along longitudinal direction.

To develop the software application, in the sense of bringing it closer to an automatic mode, it was necessary to introduce the notion of position error. This parameter characterises the difference between the desired position and current position. Because the current position is a dynamic parameter that changes in any moment (tends to desired position) it will be called instantaneous position. Thus, position error will become also dynamic parameter (tends to zero) and it will be called instantaneous position error.

Instantaneous position error must be computed in each moment (at each end time of program timer) and for each feed mechanisms. Thus, on each moment there is, for each displacement (transversal and longitudinal) an instantaneous position error. As long as one instantaneous position error, on one direction, is non-zero, the program will try to send step signal to the motor, in order to decrease that error. Attempts to transmit step signal will be done on each timer period. Not all the attempts to transmit step signal will be successfully. If it is tried to send a step signal when the output pin does not send another step, then it will start sending a signal step. Because a step signal has a length higher than specific period of program timer, it is possible that, when attempting to send a step signal another one is in progress. Since the overlap of two or more signals is not allowed, the attempt of sending one step signal during one is in progress must end with a temporary skip. Only after the current step signal is ended it can be retrieved. Thus, if there is a non-zero instantaneous position error, along one feed direction, it will be annulled as soon as possible, by moving the turning knife to the desired position. The function that determines if there is an ongoing step on longitudinal direction is:

```
function TForm1.SetPasAvansLongitudinal:boolean;
begin
if perioadaL=0 then
  begin
  dunuL:=1;
  perioadaL:=dunuL + PerioadaLong;
  SetPasAvansLongitudinal:=true;
  if b1 then PozInstLong:=PozInstLong+1
       else PozInstLong:=PozInstLong-1;
  label2.Caption:=inttostr(PozInstLong);
  end
else
  SetPasAvansLongitudinal:=false;
end;
```

For the transversal direction, the function that determines if there is an ongoing step is:

```
function TForm1.SetPasAvansTransversal:boolean;
begin
if perioadaT=0 then
  begin
  dunuT:=1;
  perioadaT:=dunuT + PerioadaTransv;
  SetPasAvansTransversal:=true;
  if b3 then PozInstTransv:=PozInstTransv+1
       else PozInstTransv:=PozInstTransv-1;
  label1.Caption:=inttostr(PozInstTransv);
  end
else
  SetPasAvansTransversal:=false;
end;
```

Procedure that manage the two instantaneous position errors, on the two feeding direction, will decide to change displacement sense and/ or to transmit a step signal corresponding to the feed mechanism. This procedure will run at the end of each period of the timer as it can be seen in the following:

```
procedure TForm1.EvaluareEroare;
begin
EroarePasL:=PozDoritaLong-PozInstLong;
EroarePasT:=PozDoritaTransv-PozInstTransv;
if   EroarePasL>0   then//PozInstLong<PozDoritaLong
then
```

```
  begin
  AvansLongitudinalDreapta;
  SetPasAvansLongitudinal;
  end
else
  if EroarePasL<0 then//PozInstLong>PozDoritaLong
then
    begin
    AvansLongitudinalStanga;
    SetPasAvansLongitudinal;
    end;
if EroarePasT>0 then// PozInstTransv<
PozDoritaTransv then
  begin
  AvansTransversalDreapta;
  SetPasAvansTransversal;
  end
else
  if EroarePasT<0 then// PozInstTransv>
PozDoritaTransv then
    begin
    AvansTransversalStanga;
    SetPasAvansTransversal;
    end;
if (EroarePasT=0) and (EroarePasL=0) then
CalcNextInput;
end;
```

When both instantaneous longitudinal and transversal position error are zero, the turning knife attains the desired position and can move on to the next desired position from a database of positions that characterise the tool trajectory.

By implementing, in the program, a procedure which calculates a list of pairs values (coordinates of points that describe the trajectory, in time, of turning tool) can move the turning knife so that it can be done complex processing. An example of such a procedure, which calculates the positions of a hundred points on the circumference of a circle placed equiangular, is shown below.

```
procedure TForm1.CalcNextInput;
begin
PozDoritaTransv:=trunc(15*sin(NrPunct*2*PI/100)*KT);
PozDoritaLong:=trunc(15*cos(NrPunct*2*PI/100)*KL);
if NrPunct<100 then NrPunct:=NrPunct+1
                    else NrPunct:=0;
end;
```

5. Conclusion

Effort to develop the CNC machine was already rewarded by the satisfaction of seeing it in action. Although multiple tests done during the lathe development decrease the importance of first processing done, that rested in leading the turning knife along a square trajectory and after that along a circle path and others. The functionality of the whole system proves that the adopted solutions where appropriate even if there were not the best. This is way the author of present paper is taken into consideration another alternative solutions that will determine the increase of positioning speed, decrease of positioning errors and development of design and execution of processing.

The lathe numerically controlled by the author of the present paper can be improved, in the future by:

- connecting two electronic limit switchers to the ends of longitudinal feed that allows zero position calibration and stroke length at start;
- connecting two electronic limit switchers to the ends of transversal feed that allows zero position calibration and stroke length at start;
- connecting an incremental transducer on longitudinal shaft to accurately detect the position of longitudinal saddle (closed loop control);
- connecting an incremental transducer on transversal shaft to accurately detect the position of longitudinal saddle (closed loop control);
- connecting an angular transducer on headstock shaft to synchronise the longitudinal, transversal and main rotation movements;
- implementing a DC motor driving for the two feed movements;
- implementing a position and a speed PID controller for both feed movements;
- implementing the whole control of the lathe on one microcontroller;
- changing the interface from parallel to USB;
- developing the application the computes data in real time;
- developing the software application that converts CNC programs done in dedicated CAD-CAM application.

References

1. Hood-Daniel, P., Floyd Kelly, J. (2009) *Build your our CNC Machine*. Springer Verlag, ISBN 978-1-4302-2490-7, New York, USA
2. Joshi, P.H. (2007) *Machine tools Handbook. Design and operation*. McGrill-Hill Publishing House, ISBN 0-07-149435-9, New York, USA
3. Zhang, X.P., Rehtanz, C., Pal, B. (2006) *Flexible AC transmission systems: modeling and control*. Springer Verlag, ISBN 978-3-540-30606-1, Berlin, Germany
4. Veltman, A., Pulle, D., DeDoncker, D. (2007) *Fundamentals of electric drives*. Springer Verlag, ISBN 978-1-4020-5503-4, Heidelberg, Germany
5. Cantu, M. (2003) *Mastering Delphi 7*. Sybex Inc., ISBN 0-7821-4201-X, Alameda, USA