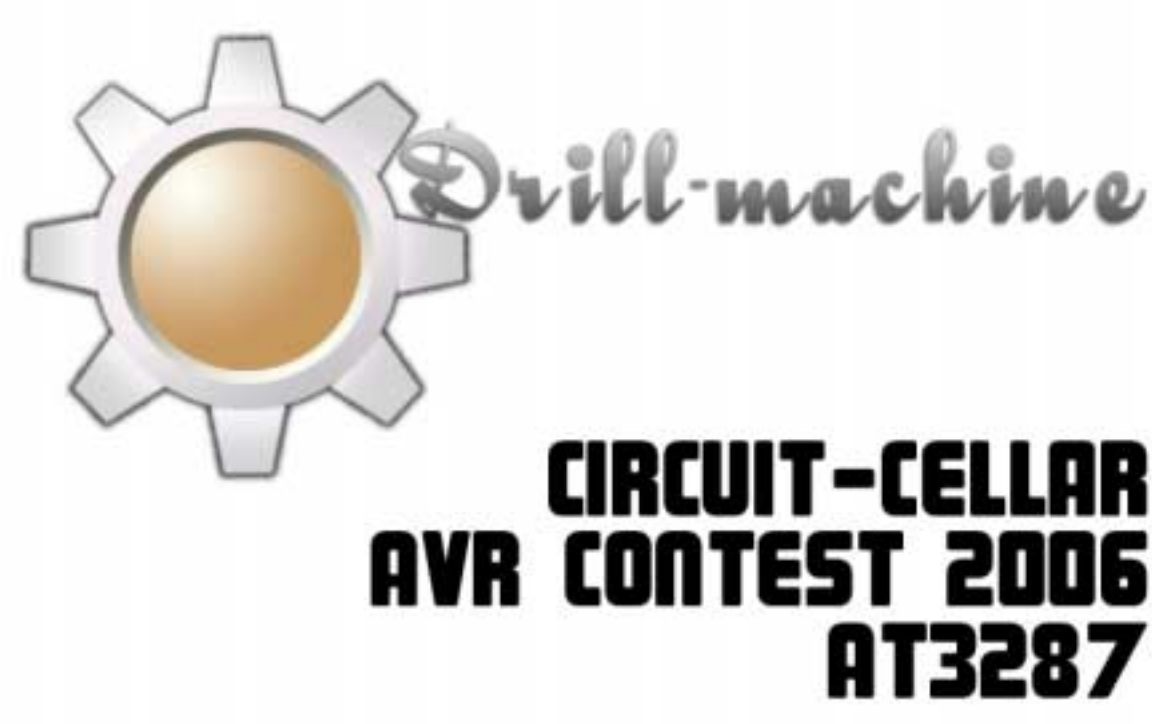
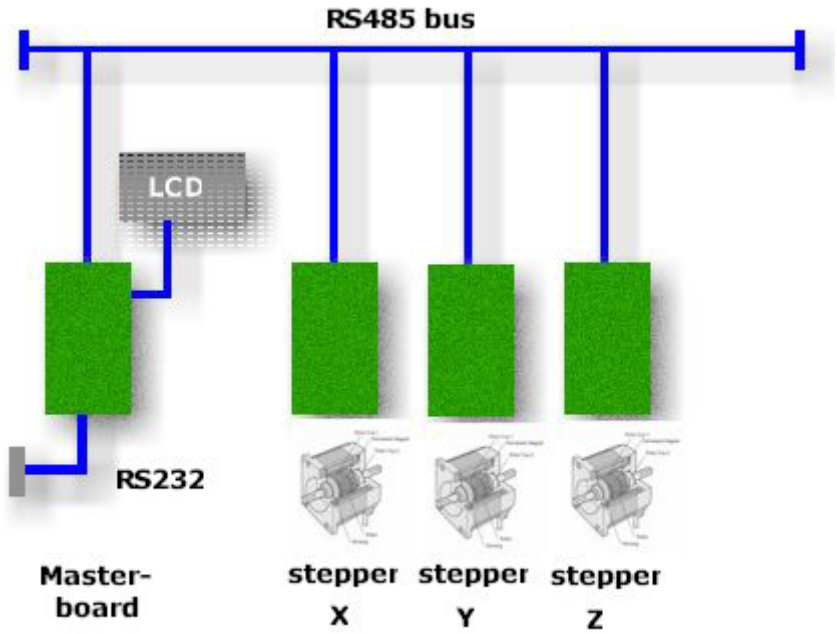


Abstract

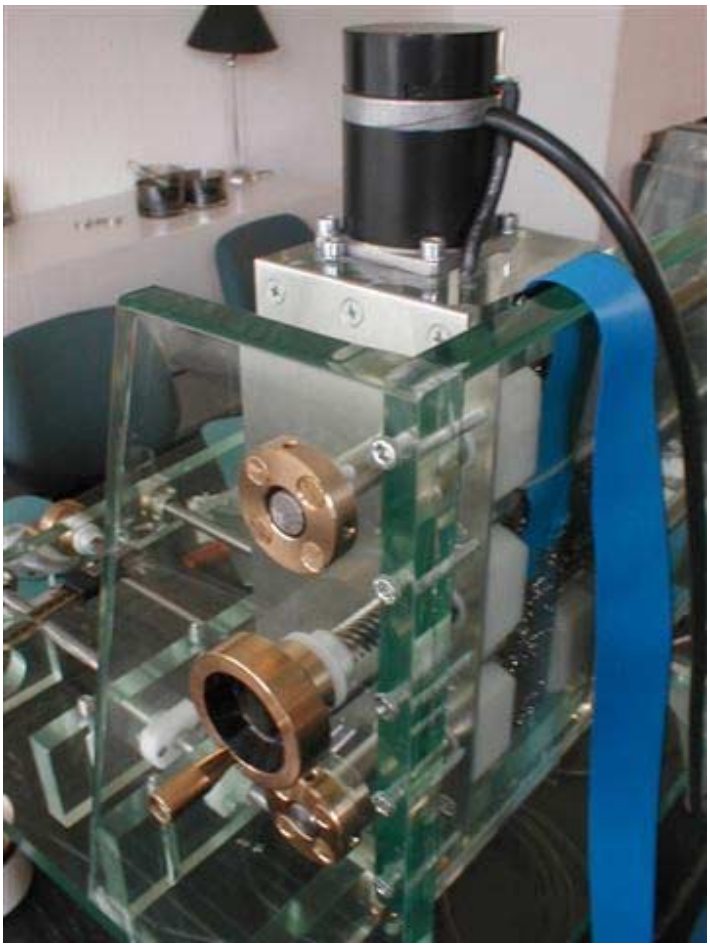
This document describes the construction of a PCB drill machine driven by a master-controller board and three stepper motor driver boards. These four single sided PC boards each contain an Atmega16/32 microcontroller. Communication between the boards is performed using the RS485 protocol with all communication initiated by the master-controller. The Atmega16/32 software is completely written in Bascom-AVR.

Blockschematics:



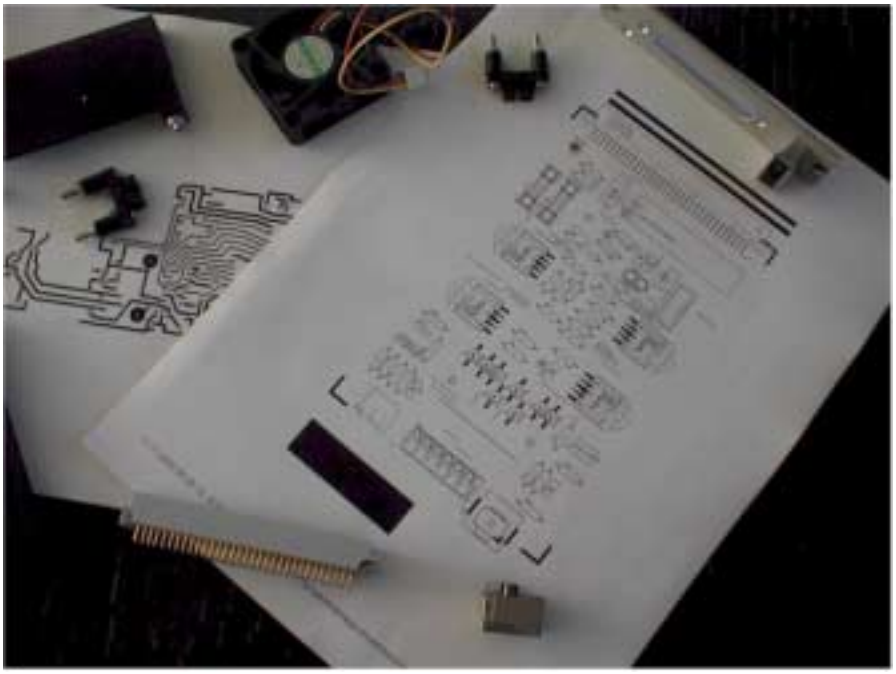
Drill-machine

HARDWARE PART 1
THE MACHINE



Drill-machine

HARDWARE PART 2
ELECTRONICS



Beginning the stepper motor driver board design.

The electronics. Master, Z-, Y- and X-board.

The boards:

On the master-board a PS2-connector to connect a PS2-keyboard. Furthermore a 8 x 24 LCD. A RJ45-connector for the RS485-bus. A Max232 for serial communication. An Atmega32 is controlling all.

On the stepper motor driver board an Atmega16. A RJ45-connector for RS485-communication. A fan to keep the temperature of the drivers for the stepper motor at a reasonable level. A stepper motor driver-board is an autonomic device. It gets a command from the master-controller-board and it takes care of the stepper motor motion, error-detection and error-correction. The initiative of communication on the RS485 bus is at the master-controller-board. It submits the stepper motor commands and will wait until the commands are done.

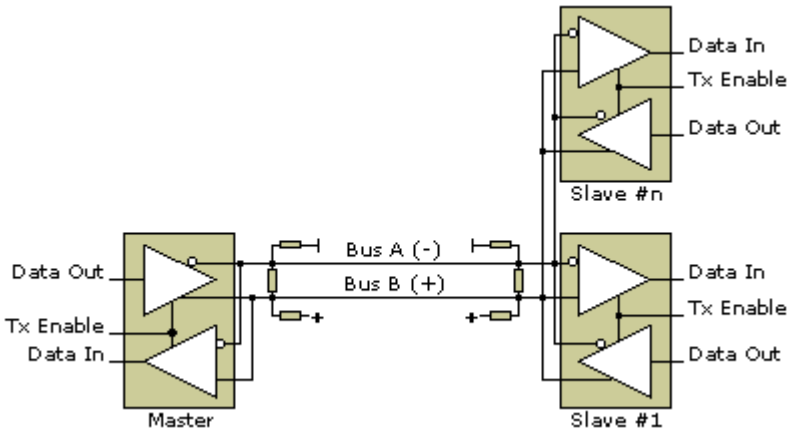
Drill-machine SOFTWARE

Completely written in Bascom-AVR.

Software for Master-board (Master)
Software for Stepper motor-driver-board (Slave X, Y and Z)

A snippet of the Bascom-AVR code for the Master.

All communication between Master board and Slave boards is done with RS485. We made up our own protocol. Destination, Source, Message, Checksum.

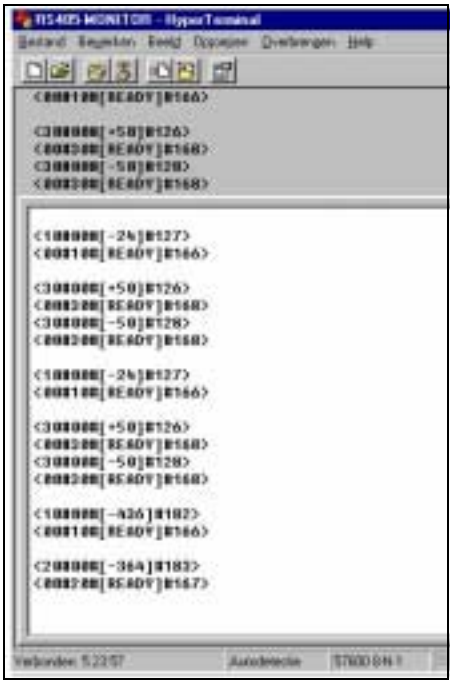


Schematics of the RS485-connection.

The initiative for communication is always done by the master-board. It sends the commands to the slaves and waits till the slave comes back with a READY or ERROR-message. Depending on the ERROR-code a resend of the command is done (slave sends an error-message because the checksum is not right), slave sends a READY when the command is finished.

For the design of printed circuit boards we have chosen Eagle. With Eagle's CAM-processor an Excellon drill-file is generated. This Excellon file is presented to the Drill-machine and the Drill-machine will do his drilling from top to bottom of the file. We haven't made any tool change equipment, but by running the Excellon file several times from top to bottom, we can use different drills. To monitor the RS485-communication-bus we added a small piece of electronics.

And now, with a communication program, like HyperTerminal, we can monitor all traffic on the RS485-communication-bus.



Here some communication-lines. This is the end of the drill-sequence. With the command <30#00#[+50]#126> followed by <30#00#[-50]#128> slave 30, the drill, is ordered to go down 50 steps and go up 50 steps. Making a hole.

A RS485-frame used on the RS485-bus.

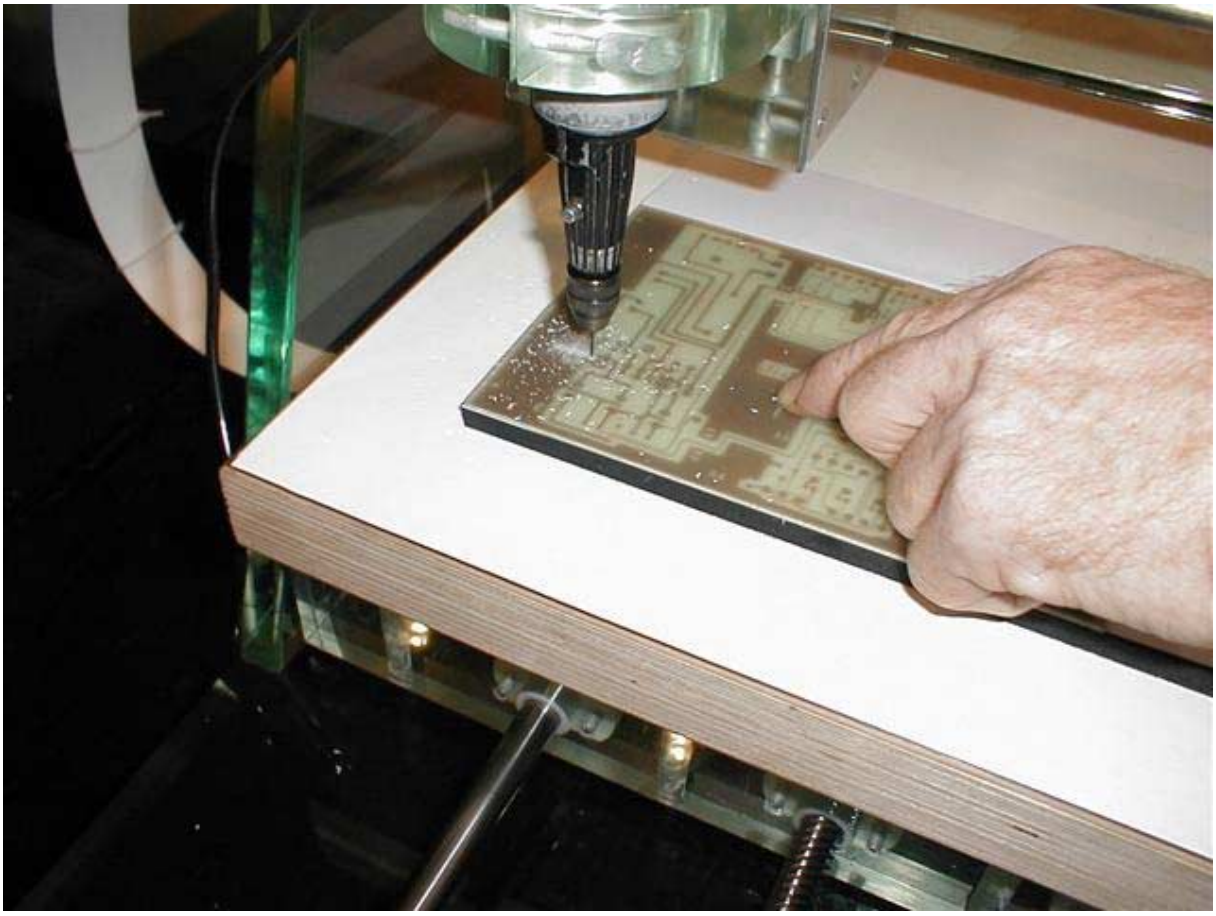
[00#10#[READY]#072]

[	**= start frame**
00	**= destination address**
#	**= separater**
10	**= source address**
#	**= separater**
[READY]	**= message/command**
#	**= separater**
072	**= checksum**
]	**= end frame**

A sample of a RS485-frame. Master has address 00, X stepper motor board has address 10, Y has 20, Z has 30. Software on all slaves is equal. In the EEPROM of each slave the address on the RS485-bus is stored. After a power up, the Z stepper motor board will move its drill to the upper limit. It will wait for a tool change (manually). The Y-slave will bring the bed to the front for mounting the PCB. And after pressing a C for “continue” on the keyboard, the Y is brought to its reference-point, and the drill is taken to his upper-drill-position. Ready to take off…..

Drill-machine

THE RESULT.....



The result. A drill-machine for PCB's. Driven by a master and three slave boards.